UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Dissertação

EXEHDA-SO: Uma Abordagem Ontológica para Ciência de Situação Aplicada ao Domínio de Segurança da Informação

Diórgenes Yuri Leal da Rosa Rosa

Pelotas, 2018

Diórgenes Yuri Leal da Rosa Rosa

EXEHDA-SO: Uma Abordagem Ontológica para Ciência de Situação Aplicada ao Domínio de Segurança da Informação

Dissertação apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Mestre em Ciência da Computação

Orientadora: Profª. Drª. Ana Marilza Pernas
Coorientador: Prof. Dr. Adenauer Corrêa Yamin

Pelotas, 2018

Insira AQUI a ficha catalográfica
(solicitada na página da biblioteca)

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

R788e Rosa, Diórgenes Yuri Leal da

Exehda-so : uma abordagem ontológica para ciência de situação aplicada ao domínio de segurança da informação / Diórgenes Yuri Leal da Rosa ; Ana Marilza Pernas, orientadora ; Adenauer Yamin Correa, coorientador. — Pelotas, 2017.

86 f.

Dissertação (Mestrado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2017.

1. Computação ubíqua. 2. Ciência de situação. 3. Segurança da informação. 4. Ontologias. 5. Processamento de contexto. I. Pernas, Ana Marilza, orient. II. Correa, Adenauer Yamin, coorient. III. Título.

CDD : 005

Elaborada por Maria Inez Figueiredo Figas Machado CRB: 10/1612

Dedico esta dissertação à minha família e amigos
por todo o incentivo e apoio.

AGRADECIMENTOS

A minha amada Daniela, por todos os momentos de incentivo, paciência e amor.

Aos meus pais Lecy e Jorge, por todo o carinho e dedicação e a minha irmã Aline, por seus constantes exemplos de força que sempre inspiram.

Aos meus orientadores, Ana e Adenauer, que foram fundamentais ao longo desta trajetória. Por todas as conversas, direcionamentos e palavras de incentivo que conduziram este trabalho.

A todos os colegas do grupo de pesquisa LUPS, em especial aos amigos Roger e Ricardo. É um privilégio contar com vocês nas inúmeras trocas de experiências e idéias.

Aos colegas de trabalho na UFPel que também deram suporte à esta importante realização da minha vida.

Ninguém caminha sem aprender a caminhar, sem aprender a fazer o caminho caminhando, refazendo e retocando o sonho pelo qual se pôs a caminhar. — PAULO FREIRE

RESUMO

ROSA, Diórgenes Yuri Leal da Rosa. **EXEHDA-SO: Uma Abordagem Ontológica para Ciência de Situação Aplicada ao Domínio de Segurança da Informação**. 2018. 86 f. Dissertação (Mestrado em Ciência da Computação) – Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2018.

As infraestruturas computacionais modernas, típicas da Computação Ubíqua, pressupõem características de flexibilidade e de permissividade quanto a conectividade do ambiente. Estas características contribuiram ao longo dos últimos anos com a concretização da emergente Internet das Coisas, a qual estende a demanda de conectividade e, por conseguinte, eleva o tráfego em redes de computadores. Entretanto, os mesmos fatores que permitem estas evoluções também potencializam problemas no que diz respeito a Segurança da Informação.

Recorrentemente são implantadas, em redes de computadores, soluções de Segurança da Informação para fins específicos, desenvolvidas em linguagens de sintaxe própria, provendo eventos em formatos também distintos. Estes fatores individualizam a análise destas soluções, o que acaba dificultando a identificação de incidentes. Neste sentido, a Ciência de Situação, enquanto estratégia capaz de integrar eventos de diferentes fontes, torna-se requisito fundamental para a implementação de controles de segurança, permitindo a flexibilidade típica da UbiComp.

Considerando isto, a presente dissertação propõe uma abordagem ontológica para Ciência de Situação aplicada ao domínio de Segurança da Informação, denominada EXEHDA-SO (*Execution Environment for Highly Distributed Applications - Security Ontology*). Por meio de processamento de eventos heterogêneos, provenientes de diferentes fontes contextuais, busca-se uma contribuição a fase de compreensão de Ciência de Situação. O modelo EXEHDA-SO é apresentado em três fragmentos denominados Core, Scope e InterCell Analyzer.

De forma a validar o modelo proposto foi desenvolvido um estudo de caso alusivo à infraestrutura computacional da Universidade Federal de Pelotas. Nesta avaliação, considerando as características de heterogeneidade e distribuição do ambiente, foi possível observar as principais contribuições propostas nesta dissertação.

Palavras-chave: Computação Ubíqua, Ciência de Situação, Segurança da Informação, Ontologias, Processamento de contexto.

ABSTRACT

ROSA, Diórgenes Yuri Leal da Rosa. **EXEHDA-SO: Uma Abordagem Ontológica para Ciência de Situação Aplicada ao Domínio de Segurança da Informação**. 2018. 86 f. Dissertação (Mestrado em Ciência da Computação) – Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2018.

Modern computing infrastructures, typical of Ubiquitous Computing, assume characteristics of flexibility and permissiveness regarding the connectivity of the environment. These characteristics have contributed over the last few years to the emerging Internet of Things, which extends the demand for connectivity and therefore raises computer networks traffic. However, the same factors that allow these evolutions also potentiate problems with regard to Information Security.

Information security solutions for specific purposes are developed in computer networks, developed in their own syntax languages, providing events in different formats. These factors individualize the analysis of these solutions, which brings difficulties to incidents identification. In this sense, Situational Science, as a strategy capable of integrating events from different sources, becomes a fundamental requirement for the security controls implementation, allowing the typical flexibility of UbiComp.

Considering this, the present dissertation proposes an ontological approach to Situation Science applied to the Information Security domain, called EXEHDA-SO (Execution Environment for Highly Distributed Applications - Security Ontology). Through the processing of heterogeneous events, coming from different contextual sources, a contribution is made to the understanding phase of Situational Science. The EXEHDA-SO model is presented in three fragments called Core, Scope and InterCell Analyzer.

In order to validate the proposed model a case study was developed allusive to Universidade Federal de Pelotas computational infrastructure. In this evaluation, considering the characteristics of heterogeneity and distribution of the environment, it was possible to observe the main contributions proposed in this dissertation.

Keywords: Ubiquitous Computing, Context Awareness, Information Security, Ontologies, Context processing.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
ATNA	*Audit Trail and Node authentication*
CMS	*Content Management System*
CA	Campus Anglo
CAPEC	*Common Attack Pattern Enumeration and Classification*
CCL	Campus Capão do Leão
CEP	*Complex Event Processing*
CS	Ciência de Situação
CSV	*Comma-separated values*
CVE	*Common Vulnerability Enumeration*
CWE	*Common Weakness Enumeration*
DMZ	*demilitarized zone*
EXEHDA	*Execution Environment for Highly Distributed Applications*
EXEHDA-USM	*Execution Environment for Highly Distributed Applications - Unified Security Management*
EXEHDA-SO	*Execution Environment for Highly Distributed Applications - Security Ontology*
ELK	*Elasticsearch, Logstash, and Kibana*
HTTP	*Hypertext Transfer Protocol*
IETF	*(Internet Engineering Task Force)*
IDS	*Intrusion detection system*
IHE	*Integrating the Health care Enterprise*
IP	*Internet Protocol*
IRI	*Internationalized Resource Identifier*
OLED	*OntoUML lightweight editor*
IoT	*Internet of Things*
ISO	*International Organization for Standardization*

LUPS	*Laboratory of Ubiquitous and Parallel Systems*
ORM	*Object Role Modeling*
OpenVAS	*Open Vulnerability Assessment System*
OWL	*Web Ontology Language*
OWL DL	*Web Ontology Language Description Logics*
OWL QL	*Web Ontology Language Query Language*
PwC	*PricewaterhouseCoopers*
RDF	*Resource Description Framework*
RDFS	*Resource Description Framework Schema*
RHIC	Repositório Híbrido de Informações Contextuais
SGBDR	Sistema Gerenciador de Banco de Dados Relacional
SIEM	*Security Information and Events Management*
SIP	*(Session Initial Protocol)*
SPARQL	*SPARQL Protocol and RDF Query Language*
SQL	*Structured Query Language*
SWRL	*Semantic Web Rule Language*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*
UFPel	*Universidade Federal de Pelotas*
UML	*Unified Modeling Language*
URI	*Uniform Resource Identifier*
VoIP	*(voice over IP)*
W3C	*World Wide Web Consortium*
WAF	*Web Application Firewall*
Wi-Fi	*Wireless Fidelity*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A Computação Ubíqua (UbiComp), definida por Mark Weiser em (WEISER, 1991), prevê diversas características e requisitos para o avanço tecnológico do presente século. Para atender as prerrogativas de UbiComp, tarefas e funcionalidades cotidianas geralmente passam a contar com algum tipo de conectividade, impondo um ambiente de redes flexível e atento a constantes mudanças. Dentre os aspectos mais significativos deste paradigma pode-se citar ainda a transparência, atributo que destaca a importância da computabilidade se fazer efetiva na vida dos seres humanos sem causar impacto, dificuldades ou barreiras. Logo, percebe-se que a ubiquidade visa aproximar ao máximo a computação daquilo que é tangível, mantendo-a imperceptível, invisível. Contemplar estas premissas, em se tratando de redes de computadores, sugere uma infraestrutura permissiva, isto é, com o menor número de bloqueios e controles possível.

As expectativas de UbiComp são observadas hoje mais especificamente na denominada Internet da Coisas, do inglês Internet of Things (IoT). IoT remete a um *framework* conceitual que visa incorporar conectividade, compartilhando dados entre múltiplos dispositivos. A adição de novas coisas na internet, independente de suas funcionalidades, trás consigo novos desafios oriundos dos seus requisitos de conectividade. Podem-se elencar diversos aspectos neste sentido como: (I) aumento de conexões simultâneas e conseguinte elevação do tráfego de rede; (II) aumento na complexidade das conexões com a utilização de diversas portas de conexão; e (III) maior exposição e frentes para ciberataques a partir de diversos métodos.

As infraestruturas computacionais derivadas da UbiComp e da decorrente IoT requerem atenção no que diz respeito às defesas necessárias às informações que trafegam nos ambientes. Neste enfoque, a Segurança da Informação (SI) se torna requisito necessário para o bom funcionamento destas infraestruturas. Apenas no Brasil foram 647.112 incidentes de segurança reportados ao Cert.br[1] no ano de 2016, observando que nem sempre os incidentes são reportados pelas organizações. Desde impactar

[1] Cert.br - Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil, mantido pelo NIC.br, do Comitê Gestor da Internet no Brasil

um concorrente comercial até obter informações estratégicas de um pais inteiro, as motivações para os constantes ataques variam sua amplitude e determinam questões econômicas e geopolíticas. Esta perspectiva é sublinhada pela pesquisa da PwC (PricewaterhouseCoopers) (PWC, 2018), onde observa-se que violações à segurança digital são cada vez mais comuns e, neste sentido, as organizações precisam tornar suas infraestruturas mais resilientes.

Observa-se dessa forma que existe um conflito entre a concretização das premissas de ubiquidade computacional e SI. Em se tratando de redes de computadores, a tentativa de flexibilizar o ambiente para torná-lo mais apto aos desafios de UbiComp pode aumentar os riscos às informações expondo vulnerabilidades. Por outro lado, a implementação de controles inadequados e sistemas de moderação de conectividade podem acarretar bloqueios e indisponibilidades atípicas à UbiComp. Neste sentido é importante que a implementação de controles técnicos considere a percepção de ataques de forma não individualizada.

Ao encontro desta indicação destaca-se a Ciência de Situação (CS), que pode ser vista como um requisito para a viabilidade da UbiComp e, concomitantemente, para a potencialização de ambientes seguros. A CS é um aspecto que fomenta a identificação de conjunturas complexas, determinando assim auxílio importante aos analistas de segurança nas tomadas de decisão para proteção da infraestrutura. Dey (1999), afirma que um sistema é ciente de contexto se este utiliza contexto para fornecer informações ou serviços relevantes para o usuário, onde a relevância depende da tarefa do usuário.

Para SI o uso de métodos voltados à CS auxilia na detecção de vulnerabilidades, ataques ou qualquer outro tipo de incidente de segurança. Eventos de segurança são fontes importantes para que, devidamente relacionados, possam prover CS no mutável e dinâmico domínio de SI evitando assim uma minimização desnecessária da aptidão do ambiente à flexibilidade.

Posto este alinhamento, esta dissertação busca propor uma solução em CS por intermédio de uma estratégia baseada no uso de ontologias. Estas estruturas proveem entendimento compartilhado e processável sobre um domínio de conhecimento, podendo especificar relações semânticas entre diferentes situações e identificando cenários de alto nível. A estratégia ontológica também adequa-se a SI por ser capaz de contemplar diversas fontes de informação contextual, unificando formatos.

1.1 Motivações e Objetivos

Considerando as questões mencionadas anteriormente, destacam-se como problemas específicos, que motivam esta dissertação:

- o uso recorrente de soluções de propósito específico no domínio de SI dificulta

a identificação de incidentes de segurança, devido aos diferentes formatos de eventos produzidos por cada uma destas;

- as atuais soluções usadas para correlação de eventos em SI costumam utilizar linguagens com sintaxes próprias, o que dificulta sua reutilização e não propicia aproveitamento compartilhado das constantes evoluções nas regras de correlação;

- o emprego de soluções distintas para fins específicos de SI trás empecilhos quanto a integração de diferentes contextos, não permitindo uma visão unificada do ambiente.

Desta forma, o objetivo central deste trabalho é empregar uma estratégia ontológica para processamento de eventos na etapa de compreensão de CS no domínio de SI. O trabalho contribui com a abordagem previamente desenvolvida pelo grupo de pesquisa, denominada *Execution Environment for Highly Distributed Applications - Unified Security Management* (EXEHDA-USM) (ALMEIDA, 2016), que visa o fornecimento de CS sobre aspectos relacionados à SI por meio de uma arquitetura hierárquica multinível.

Como objetivos específicos, destacam-se:

- conceber um modelo ontológico que contemple conceitos básicos de SI e informações sobre a arquitetura e os ativos que suportam as funcionalidades de um ambiente típico em UbiComp;

- integrar o modelo aos conceitos já consolidados pelos trabalhos prévios do grupo de pesquisa LUPS (Laboratory of Ubiquitous and Parallel Systems) (LOPES et al., 2014) e (ALMEIDA, 2016);

- implementar e testar raciocínio ontológico em ambiente simulado alusivo à arquitetura prevista no EXEHDA-USM e, por conseguinte, ao ambiente computacional da UFPel (Universidade Federal de Pelotas).

1.2 Estrutura do Texto

Esta dissertação divide-se em 7 Capítulos onde o Capítulo seguinte trata da teoria de CS com um enfoque em SI. Ao visualizar as etapas necessárias à CS busca-se traçar um paralelo à etapa de compreensão e ao uso de ontologias, eixo central desta dissertação.

Logo, no Capítulo 3, são retomados conceitos importantes do projeto EXEHDA-USM, projeto no qual este trabalho constitui uma contribuição. Também são tratadas características do conjunto tecnológico típico às estratégias ontológicas.

O Capítulo 4 traz uma amostragem de trabalhos relacionados às verticais principais deste trabalho. Destes também foram reutilizados diversos conceitos e relações de forma adaptada a EXEHDA-SO.

O quinto Capítulo mostra a proposta ontológica desenvolvida, bem como a estratégia que foi utilizada para prover CS no âmbito de SI. Também visa demonstrar como o modelo integra-se à trabalhos prévios do grupo de pesquisa.

O Capítulo 6 descreve cenários nos quais a estratégia foi aplicada no intuito de validar a proposta apresentada. Os cenários fazem uso de um ambiente simulado aluzivo ao parque computacional da UFPel.

O sétimo e último Capítulo conclui o trabalho discutindo as principais contribuições e resultados, trazendo as oportunidades de trabalhos futuros e os artigos publicados.

2 CIÊNCIA DE SITUAÇÃO APLICADA A SEGURANÇA DA INFORMAÇÃO

A segurança da informação (SI) detém papel de elevada relevância dentre as características que compõem a UbiComp e, por conseguinte, na contemporânea internet das coisas, do inglês *Internet of Things* (IoT) (KUMAR; PATEL, 2014; ATZORI; IERA; MORABITO, 2010). Há, no âmbito de tecnologia da informação (TI), voltada ao suporte IoT, uma recorrente busca pelo equilíbrio entre dinamicidade e proteção das infraestruturas, de forma que os sistemas e os ambientes contenham o maior nível de proteção possível, sem que isto impacte a operação das funcionalidades destes ativos, observadas as constantes mudanças e os diferentes cenários.

Para que esta expectativa seja contemplada, comumente são implementadas diversas ferramentas de redes, como *firewall's*, *Intrusion Detect Systems* (IDS's), *Web Application Firewall* (WAF's), entre outros. Contudo sem uma constante avaliação sobre as métricas obtidas por essas ferramentas, bem como do tráfego que está sendo gerado, os controles implementados podem não alcançar o êxito esperado. Neste sentido, um dos principais requisitos teóricos para que os mecanismos de segurança sejam efetivos recai justamente sobre a denominada Ciência de Situação (CS).

Existem hoje uma série de abordagens para a modelagem de contexto sendo utilizadas de acordo com as especificidades de cada aplicação. Dentre estas abordagens, os mais simples são os modelos chave-valor onde o contexto é representado meramente por atributo e valor correspondente. Já nos modelos baseados em linguagens de marcação há uma evolução na estratégia que faz uso de estruturas de dados hierárquicas, consistindo de *tags* com atributos e conteúdo para armazenamento de contexto. Os modelos gráficos utilizam grafos contextuais, ORM (*Object Role Modeling*) e UML (*Unified Modeling Language*).

Dentre estas, visando modelagem de contexto, destaca-se o uso de ontologias, por contemplar o trabalho com composição de contexto distribuída, validação parcial, incompleta ou ambígua. Também por possuir uma sintaxe bem definida e por ser facilmente aplicada a cenários de aplicação já em produção (STRANG; LINNHOFF-POPIEN, 2004).

Posto este alinhamento, no qual se baseia o presente trabalho, este capítulo traz inicialmente uma revisão conceitual acerca de Ciência de Contexto e Situação. Na sequência, é relacionada a importância da ciência situacional à obtenção de segurança da informação (SI). E como uma das técnicas de reconhecido destaque voltada para conhecimento situacional em ambientes computacionais é abordado o uso de ontologias, tema que encerra o conteúdo do Capítulo demonstrando sua aplicabilidade no domínio abordado.

2.1 Ciência de Contexto

Entende-se por contexto toda a informação que possa ser usada para caracterizar uma determinada situação (DEY, 2001). Esta informação pode ser oriunda das mais diversas fontes, desde o mais simples sensor binário, até elaborados arquivos de logs com inúmeros campos e/ou extensos registros em bancos de dados, entre outros. Os dados em questão referem-se sempre a uma determinada entidade, podendo ser por exemplo uma pessoa, um local, um objeto, desde que possam ser caracterizados e contenham relevância para funcionalidades desenvolvidas para o ambiente. Define-se que Ciência de Contexto é a capacidade dos sistemas de informação e ambientes em contemplarem circunstâncias diversas, correlacionando-as e compreendendo-as. O trabalho (SCHMIDT; BEIGL; GELLERSEN, 1998) identifica Contexto como uma questão chave na interação entre humanos e computadores, descrevendo os fatos que cercam esta relação e acrescentando significado. No trabalho de (COUTAZ et al., 2005), por sua vez, afirma que Contexto não é simplesmente um estado predefinido de um ambiente com um conjunto fixo de recursos de interações, mas sim parte de um processo de interação com um ambiente de constante mudança composto por recursos distribuídos.

Em síntese, a construção de um sistema ciente de contexto, segundo (BERNARDOS; TARRIO; CASAR, 2008) se dá por meio de três etapas, sendo elas: aquisição, modelagem e processamento. A figura 1, extraída de (PERERA et al., 2013a), contempla abordagem semelhante apresentando um exemplo do ciclo de vida do contexto que também perpassa as etapas de aquisição, modelagem, raciocínio (equivalente a processamento) somando-se a etapa de disseminação.

Em se tratando da categorização acerca de contexto, pode ser feita por diversos pontos de vista. Por exemplo, (J. PASCOE; MORSE, 1998) aponta questões como localização, ambiente, identidade, tempo e (DEY; ABOWD, 1999) acrescenta atividade suprimindo ambiente. Pode-se verificar que estas categorias trazem uma visão sobre como, onde, quando e o que aconteceu, possivelmente levando também a uma causa. Estas maneiras de encarar contexto integram-se às demandas de SI e são mencionadas na literatura frequentemente como 5W2H, do inglês *what, why, where, when, who,*

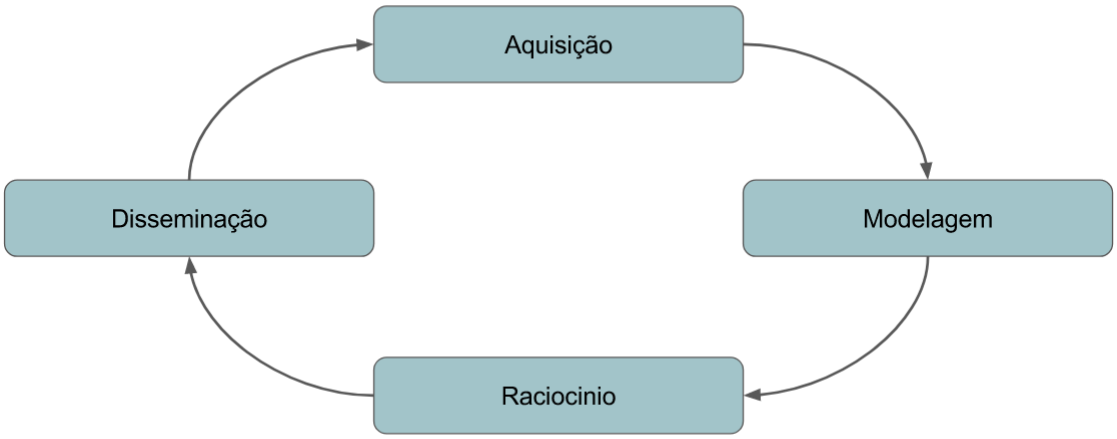

Figura 1: Ciclo de vida de construção e operação de um sistema ciente de contexto
Fonte: (PERERA et al., 2013b)

how e *how much*. Um exemplo desta classificação é apresentado na seção 2.2.1. A seguir serão visitadas as fases do ciclo de vida do contexto.

2.1.1 Aquisição

Vários aspectos quanto à aquisição de contexto podem ser destacados (PERERA et al., 2013b), como por exemplo quanto a(ao):

- responsabilidade: refere-se a aquisição que pode ocorrer ativamente por requisições dos componentes de software ou passivamente, aguardando as entradas dos sensores
- frequência: de forma instantânea ao evento ou de maneira intervalar com um tempo determinado entre as aquisições
- fonte: aquisição ocorrendo por intermédio de sensores, por um *middleware*, por um servidor de contexto
- tipo de sensores: físicos, virtuais ou lógicos
- processo de aquisição: sensorial, derivada (gerada por operações computacionais sobre os dados sensoriados) ou provida manualmente

2.1.2 Modelagem

O processo de modelagem de contexto consiste na concepção de um modelo de entidades do mundo real, suas propriedades, estado de seu ambiente e situações que podem ser usadas como referência para a aquisição, interpretação e raciocínio de

informações contextuais (KNAPPMEYER et al., 2013). Um bom formalismo para modelagem de contexto reduz a complexidade das aplicações cientes ao contexto, facilita o acesso às informações realizando buscas de forma eficiente, melhora a capacidade de manutenção e de evolução da aplicação (STRANG; LINNHOFF-POPIEN, 2004). Alguns requisitos que devem ser levados em consideração durante a modelagem das informações de contexto são: heterogeneidade, mobilidade, dependências e relações entre os dados.

A realização de modelagem de contexto pode ocorrer de diversas maneiras já estabelecidas como modelo chave-valor, modelo baseado em linguagens de marcação; modelos gráficos; orientados a objetos; modelos baseados em lógica e também modelos baseados em ontologias. Uma comparação entre estas técnicas é apresentada em (STRANG; LINNHOFF-POPIEN, 2004) onde as ontologias levam vantagem em diversos aspectos importantes para o domínio delimitado para este trabalho, tais como composição distribuída e parcial de contexto. Esta comparação é fortalecida no trabalho (BETTINI et al., 2010), o qual também apresenta as ontologias como sendo mais adequadas para modelagem contextual. Considerando isto e o foco deste trabalho que, entre outros aspectos, visa a aplicação de ontologias na etapa de modelagem de contexto, características e conceitos principais do desenvolvimento e aplicação de ontologias são ampliados na seção 2.3.

2.1.3 Raciocínio

A etapa de raciocínio considera as diversas detecções do ambiente relacionando-as afim de gerar determinado conhecimento, podendo efetuar inferências de alto nível avaliando informações de baixo nível (PERERA et al., 2014) . Esta etapa determina, observados parâmetros estabelecidos previamente, quais são as adaptações necessárias em um ambiente ubíquo. Há então uma tomada de decisão que será concretizada na fase de disseminação. Dentre as diversas formas de executar o raciocínio e gerar inferências, segundo (PERERA et al., 2014), o método de regras é o mais explorado, o qual alinha-se com a estratégia adotada neste trabalho.

2.1.4 Disseminação

A disseminação diz respeito à maneira como o contexto será entregue aos interessados provendo métodos para esta entrega. Os sistemas utilizam diversos destes métodos como, por exemplo, (i) consultas, onde os consumidores executam sua solicitação de contexto ou ainda (ii) subscrição, onde o sistema entrega o contexto aos subscritos em uma abordagem denominada *publisher/subscriber*. Pode-se afirmar ainda que o ambiente computacional deve ser adaptado enquanto a situação está ocorrendo, utilizando valores atuais de sensores, bem como informações históricas dos mesmos (YE; STEVENSON; DOBSON, 2011).

2.2 Ciência de Situação

Ciência de Situação (CS) consiste na caracterização do contexto de diversas entidades relevantes do ambiente. O trabalho (KOKAR; MATHEUS; BACLAWSKI, 2009) apresenta uma abordagem sobre CS e é ilustrada pela figura 2, onde podem-se identificar 4 camadas.

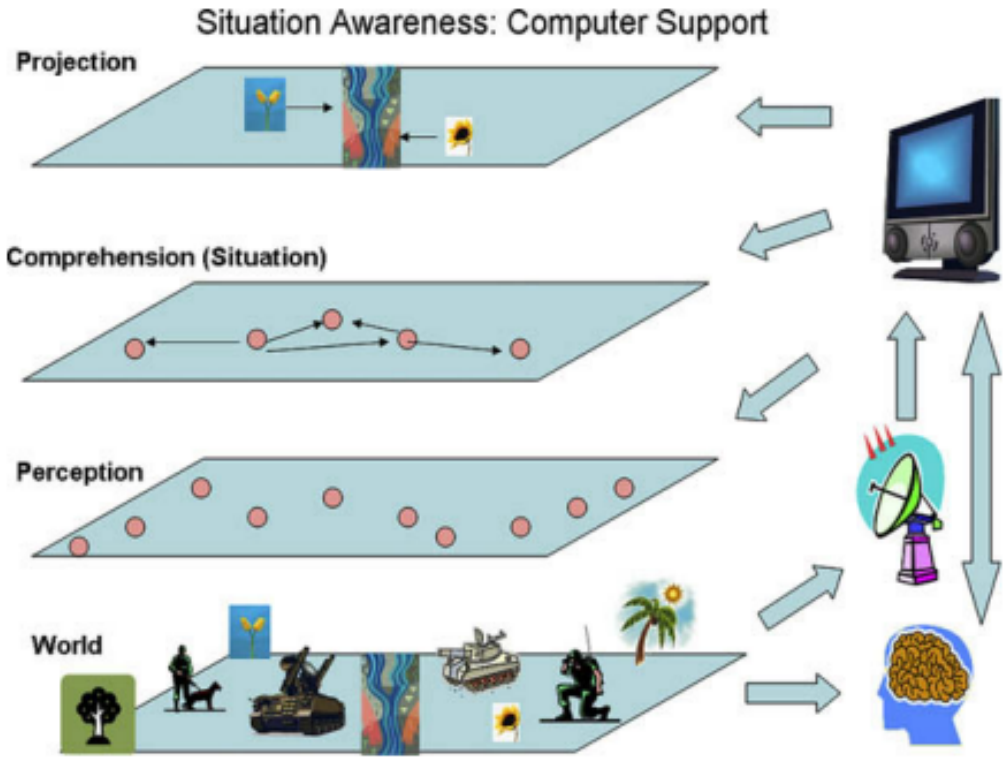

Figura 2: Camadas de Ciência de Situação
Fonte: (KOKAR; MATHEUS; BACLAWSKI, 2009)

Na camada inferior apresenta-se uma representação do mundo físico, isto é, uma abstração para a atuação de possíveis entidades componentes do ambiente. Ao lado direito do plano pode-se observar que há a representação da correlação existente entre o raciocínio humano e a CS. Onde antes existia apenas o entendimento contextual humano do mundo, agora tem-se a computação provendo também contexto e fazendo uso de entradas de dados e informações humanas de contexto.

A segunda camada consiste da Percepção. Os pontos no plano representam computacionalmente os objetos monitorados por sensores. O processo sensorial é pontuado na figura pela flecha que liga a camada do mundo físico a uma espécie de radar que obtém e repassa os dados. É possível observar que esta camada de Percepção é responsável pela normalização dos dados recebidos.

Na camada seguinte está a abstração para Compreensão onde é possível ver a

representação de possíveis correlações existentes entre as entidades. Esta camada refere-se a como combinar, interpretar e armazenar os dados recebidos. Uma correta compreensão dos dados evidencia a melhor técnica de modelagem de contexto a ser utilizada dentre as várias existentes.

Finalizando a análise da figura 2, a camada superior é denominada Projeção. Esta camada de abstração é responsável por inferir prognósticos, levando em conta a compreensão dos eventos ocorridos.

2.2.1 Ciência de Situação na Segurança da Informação

Em (BASS, 2000) Tim Bass propôs a aplicação dos conceitos de CS no campo da segurança em redes de computadores, visando o fornecimento de uma visão mais aprimorada. O autor é tido como o primeiro a empregar CS na obtenção de um melhor entendimento sobre o ambiente computacional.

A correlação existente entre as duas áreas destacadas pode ser observada pelo conceito de *Network Security Situational Awareness* (NSSA), tratando-se de uma terminologia usada para descrever CS na defesa de redes de computadores. NSSA busca proporcionar conhecimento e possibilidades de análise de segurança, percepção de situações de risco, otimizando os processos de tomadas de decisões, podendo prover inclusive predições para estados em ambientes de complexidade elevada (ONWUBIKO, 2009).

Conforme destacado anteriormente o contexto pode muitas vezes ser trabalhado pelo conceito 5W2H, onde para o domínio de SI pode-se exemplificar a categorização da seguinte maneira:

- *What:* o que ocorreu no ambiente ubíquo ou quais foram as funcionalidades afetadas
- *Why:* qual a motivação do atacante
- *Where:* de onde e para onde está ocorrendo a situação
- *When:* quando ocorreu e quando foi detectada a inconformidade
- *Who:* qual dispositivo ou usuário executou a ação
- *How:* qual método foi utilizado
- *How much:* qual será o impacto (financeiro ou não) de um possível sucesso de determinada ação maliciosa

É importante observar ainda que a SI pode ser considerada uma área abrangente quanto as formas de ataques e vulnerabilidades existentes. A construção de ambientes computacionais seguros demanda cuidados desde a fase de requisitos de sistema,

passando pelo estabelecimento de *hardening* em servidores e até treinamento de pessoas. A representação do conhecimento e relações possíveis para conceitos que são oferecidas pelas ontologias possuem a potencialidade de auxiliar o analista no acompanhamento dos diversos detalhes envolvendo SI em uma organização. A seção a seguir destaca alguns aspectos iniciais sobre ontologias.

2.3 Ontologias

Ontologias provêm representação computacional de conhecimento, apresentando conceitos e ideias do domínio de aplicação e indicando relações entre eles. Segundo (GRUBER, 1993) uma ontologia é a especificação de um vocabulário para um determinado domínio, que relaciona-se diretamente ao âmbito dinâmico das redes de computadores e suas vulnerabilidades.

Dentre as razões para se desenvolver ontologias podem-se destacar: (i) a capacidade de alcançar entendimento compartilhado da informação estruturada que pode ser fundamentada e analisada automaticamente entre seres humanos e agentes de software; (ii) a habilidade de especificar várias relações semânticas entre diferentes conceitos; (iii) a potencialidade em solucionar questões de interoperabilidade, visto a heterogeneidade tecnológica atual em termos de software e hardware, e; (iv) a reusabilidade e habilidade em evoluir ao longo do tempo (USCHOLD; GRUNINGER, 1996; VOROBIEV; BEKMAMEDOVA, 2010).

As ontologias podem ser tipificadas da seguinte maneira (ISOTAMI SEIJI ; BITTENCOURT, 2015):

- Ontologias Genéricas (alto-nível): descrevem conceitos gerais, tais como espaço, tempo, matéria, objeto, evento, ação, que são independentes de um domínio particular.
- Ontologias de Domínio: descrevem um vocabulário relacionado a um domínio genérico, expressando conceitualizações de domínios particulares.
- Ontologia de Tarefas: descrevem conceitos relacionados a tarefas ou atividades genéricas, independentes do domínio em que ocorram.
- Ontologias de Aplicação: descrevem conceitos que dependem tanto de um domínio específico como de uma tarefa específica, e geralmente é uma especialização de ambos. Estes conceitos frequentemente correspondem a papéis desempenhados por entidades do domínio quando da realização de uma certa tarefa.

Uma das questões mais importantes ao se desenvolver uma ontologia é justamente a metodologia a ser empregada. A metodologia denominada *Ontology Development*

101 trazida pelo trabalho (NOY; MCGUINNESS, 2001), utilizada no desenvolvimento deste trabalho, lista as fases necessárias a criação de uma ontologia, são elas: (i) determinar o domínio e o escopo da ontologia; (ii) investigar o reuso de ontologias existentes; (iii) listar termos importantes; (iv) definir classes; (v) identificar hierarquia de classes; (vi) definir restrições e axiomas das classes. Em se tratando do domínio no qual se concentra este trabalho, que apresenta constante alteração nos nodos do ambiente, é interessante o uso deste ciclo de metodologia que permite a adequação das funcionalidades de acordo com a realidade do tráfego em questão.

As situações, conforme entendidas nesta seção, são entidades genuinamente ontológicas (COSTA et al., 2006), e a próxima seção aborda justamente a relação existente entre ontologias e situações envolvendo SI.

2.4 Aplicabilidade de Ontologias na Segurança da Informação

A área de segurança computacional tem se voltado para ontologias como subterfúgio para diversos desafios. Ao considerar que SI permeia diversos níveis tecnológicos é possível destacar o uso de ontologias atendendo os mais diversos aspectos deste domínio. O trabalho (MYLOPOULOS et al., 1990), já identificava a potencialidade de sua proposta ontológica (Telos) para o desenvolvimento de modelos de especificações de segurança. O uso de ontologias nos domínios de SI é defendido por prover: (i) um entendimento compartilhado da informação estruturada que pode ser fundamentada e analisada automaticamente entre seres humanos e agentes de software; (ii) reuso de conhecimento facilitando a evolução das soluções; (iii) melhora na questão da interoperabilidade, pois diferentes aplicações podem valer-se dos conceitos e relacionamentos definidos nas ontologias e (iv) a habilidade de especificar várias relações semânticas entre diferentes conceitos (MARTIMIANO; MOREIRA, 2006) (VOROBIEV; BEKMAMEDOVA, 2010).

(SOUAG; SALINESI; COMYN-WATTIAU, 2012) traz uma classificação abrangente da aplicabilidade de ontologias no que tange SI pontuando os campos de pesquisa mais dominantes. Como forma de ilustrar esse panorama, a figura 3 destaca os 8 referidos campos, que são ampliados a seguir:

- *Beginning Security Ontologies:* pontuando as ontologias pioneiras no domínio de SI e iniciando a classificação proposta em (SOUAG; SALINESI; COMYN-WATTIAU, 2012). Destaca-se o trabalho (MYLOPOULOS et al., 1990), que já nos anos 90, mesclava uma base de conhecimento com gerenciamento de sistemas propondo a linguagem Telos.

- *Security Taxonomies:* refere-se a uma das demandas mais pertinentes ao uso de ontologias que é o compartilhamento de conceitos e informações básicas a

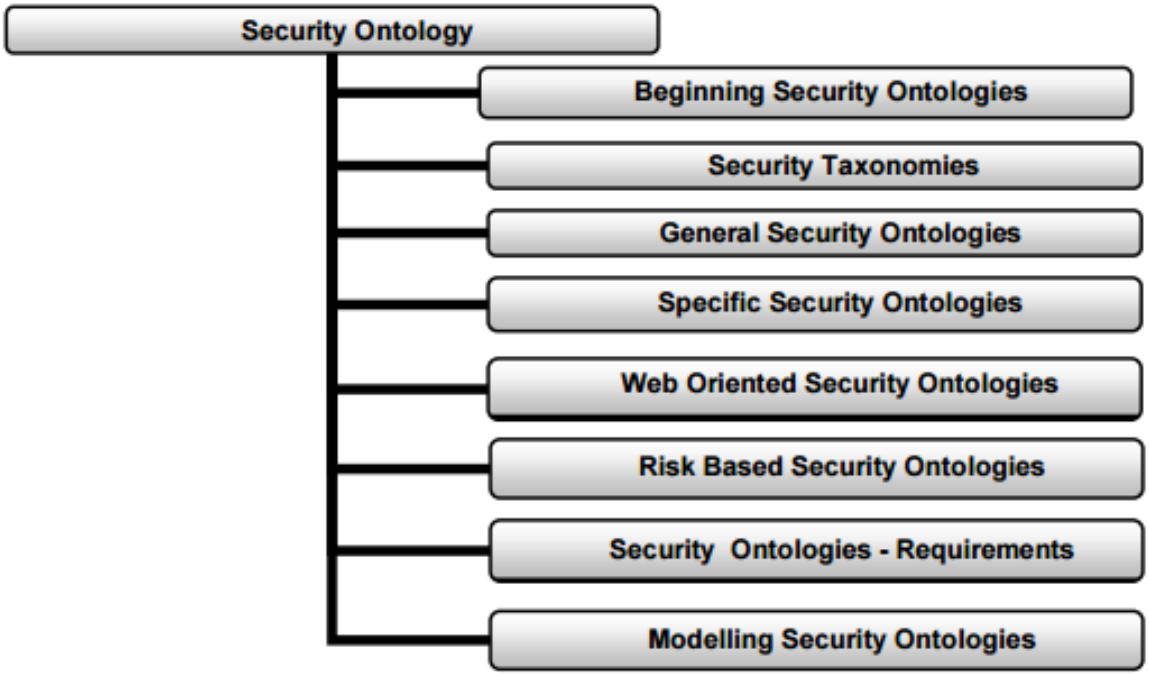

Figura 3: Classificação da aplicabilidade de ontologias voltadas para SI
Fonte: (SOUAG; SALINESI; COMYN-WATTIAU, 2012)

respeito de uma determinada área de conhecimento. No caso de SI faz-se fundamental o estabelecimento de classes que refiram-se aos conceitos fundamentais e já amplamente discutidos de forma teórica. Aqui pode haver tanto conceitos mais técnicos quanto conceitos mais amplos como a definição de uma política que coordene as ações administrativas sensíveis da organização.

- *General Security Ontologies:* estas ontologias são entendidas como aquelas que buscam contemplar todos os aspectos de SI, ampliando um pouco os conceitos básicos, estendendo a ontologia com mais vocabulário e detalhes.

- *Specific security ontologies:* enquadram-se como ontologias de segurança específicas aquelas que contemplam um cenário menor dentro do escopo de SI, aplicando suas funcionalidades à determinada particularidade do ambiente. Por exemplo, em (GENEIATAKIS; LAMBRINOUDAKIS, 2007) uma ontologia é proposta para SIP *(Session Initial Protocol)*, um protocolo especificado pela IETF *(Internet Engineering Task Force)* voltado para VOIP *(voice over IP)*.

- *Web oriented security ontologies:* atualmente as aplicações web são um dos alvos preferidos dos atacantes. Isto é demonstrado pelo CERT.br[1] que entre

[1]Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil

2013 e 2014 registrou um aumento de 54% nas notificações de ataques a servidores web, totalizando 28.808 notificações. Neste sentido as ontologias desta classificação auxiliam a segurança em aplicações web. No trabalho (VOROBIEV; HAN, 2006) são descritas ameaças em *web services* e *status* que necessitam de análise e classificação sistemática para prover o desenvolvimento de mecanismos de defesa distribuídos. O trabalho escolhe OWL (Ontology Web Language) para a taxonomia necessária, linguagem que tem sua discussão retomada no Capítulo 4.

- *Risk Based Security Ontologies:* esta metodologia destaca a importância da análise de risco para a defesa de perímetros de redes. O artigo (EKELHART et al., 2007) apresenta uma ontologia onde cada cenário de ameaça pode ser simulado com diferentes perfis de segurança para que ocorra uma avaliação da efetividade e do custo benefício das estratégias de proteção.

- *Security Ontologies - requirements:* A padronização dos controles de segurança da informação é um requisito fundamental para a concepção dos sistemas em ambientes ubíquos. O modelo arquitetural de software, por sua vez, deve ser sedimentado por um levantamento de requisitos de segurança que responda a sua criticidade. Contudo não raras vezes estas questões são preteridas em detrimento à prazos, funcionalidades finais. O trabalho (LASHERAS et al., 2009), que será abordado na seção de trabalhos relacionados, tem justamente o caráter de auxílio no levantamento de requisitos de segurança para desenvolvimento de aplicações web.

- *Moddeling Security Ontologies:* em (SOUAG; SALINESI; COMYN-WATTIAU, 2012) ainda é destacada a classificação de ontologias de segurança, que descrevem modelos, exemplificando pelos trabalhos (MOURATIDIS; GIORGINI; MANSON, 2003) (que propõe o TROPOS) e (MASSACCI et al., 2011), como trabalho que utilizam ontologias para modelagem de situações por intermédio de relações e proposições.

Desta forma pode-se perceber que a utilização de ontologias no âmbito de SI perpassa muitos aspectos. Uma observação importante é o uso de ontologias representando políticas de segurança, questão que detêm central importância para a área. Normas técnicas de segurança como as definidas por padronizações internacionais, como a ISO 27002, também são uma fonte de orientações importante que podem enriquecer e potencializar a atuação das ontologias na SI. As ontologias (AZEVEDO et al., 2008), (MARTIMIANO; MOREIRA, 2006), entre outras, tem justamente a definição de políticas de segurança como plano de fundo para muitas funcionalidades do projeto desenvolvido, o que mostra a relevância de representar computacionalmente a política

da organização também, isto é, ambientes distintos requerem diferentes níveis de controles observada a criticidade das informações que trafegam no ambiente em questão.

2.5 Considerações Finais

O capítulo buscou trazer conceitos referentes a contexto e CS, bem como a importância do levantamento de informações situacionais no âmbito de SI. Nota-se que, mesmo com o desenvolvimento dos estudos acerca de ciência de situação, ainda há muito a ser explorado e abordado em decorrência de diversos fatores, como por exemplo:

- a elevada heterogeneidade das fontes provedoras de informações contextuais.
- as diversas formas de relacionamentos possíveis entre dos dados contextuais.
- os diversos formatos que os dados contextuais podem assumir bem como o carácter polissêmico que um único evento de segurança pode ter.
- a ampla demanda por ciência de situação em áreas como SI, onde as técnicas de ataques se diversificam à medida que novos recursos são disponibilizados.

Dentre os conceitos apresentados neste Capítulo observa-se que a CS pode ser entendida como um estágio final de um processo que começa pela aquisição de dados contextuais e compreensão dos mesmos. Na SI os dados contextuais podem advir das mais diversas fontes como logs, bancos de dados, tráfego de rede, etc. A CS é também uma decorrência da compreensão e muitas vezes do raciocínio sobre os dados coletados e uma das técnicas trabalhadas para gerar essa compreensão são as ontologias. Um dos principais intuitos quando se desenvolve uma ontologia consiste em tornar o conhecimento do mundo real processável por máquina. Este fator traduz a relação existente entre a teoria de CS e ontologias.

A aplicabilidade de ontologias em SI apresentada nesta seção buscou demonstrar, de maneira abrangente, a importância da técnica para o domínio em questão e os motivos pelos quais optou-se por esta técnica para a elaboração da estratégia detalhada no Capítulo 5.

3 ESCOPO DO TRABALHO

Como base introdutória para diversos conceitos considerados neste trabalho, o presente Capítulo inicialmente destaca a arquitetura celular e componentes do EXEHDA-USM *(Execution Environment for Highly Distributed Applications - Unified Security Management)* (ALMEIDA, 2016), bem como alguns aspectos referentes ao seu precursor, o *middleware* EXEHDA *(Execution Environment for Highly Distributed Applications)*(YAMIN, 2004). Na sequência é abordado o uso de ontologias enquanto alternativa para a fase de compreensão de CS. É apresentado ainda um levantamento referente as soluções utilizadas especificamente no desenvolvimento da ontologia na seção 3.2, bem como outras soluções que apoiaram a prototipação dos estudos de caso deste trabalho, na seção 3.3.

3.1 EXEHDA-USM

De maneira introdutória ao EXEHDA-USM traz-se uma discussão sobre alguns aspectos do *middleware* EXEHDA que constituem base para grande parte do trabalho voltado a CS desenvolvido ao longo dos últimos anos pelo grupo de pesquisa LUPS *(Laboratory of Ubiquitous and Parallel Systems)*.

Na figura 4 é possível visualizar a atuação de alguns dos principais componentes estruturais do EXEHDA. Aos equipamentos de TI que valem-se do ambiente ubíquo para executar as aplicações dos usuários finais é dada a nomenclatura *EXEHDAnodos*. Desde que disponham pelo menos de uma interface de rede, os *EXEHDAnodos* podem ser desde computadores com hardware de maior capacidade, passando por dispositivos com elevada portabilidade, como celulares (EXEHDAnodo móvel), ou até mesmo sensores com poder de processamento mais restrito. O *EXEHDAbase* é visto como um servidor de convergência para a comunicação dos *EXEHDAnodos*, ele provê os serviços disponíveis ao ambiente e, dependendo da densidade de conexões, pode ser distribuído em diversos equipamentos. Na figura 4 é possível verificar a presença de Servidores de Borda (SB's) que constituem o *EXEHDAbase* e disponibilizam as aplicações do ambiente. O *EXEHDAcel*, por sua vez, denota a área de atuação de

um *EXEHDAbase*. Para definir a abrangência de uma *EXEHDAcel* são considerados o escopo institucional, a proximidade geográfica e o custo de comunicação.

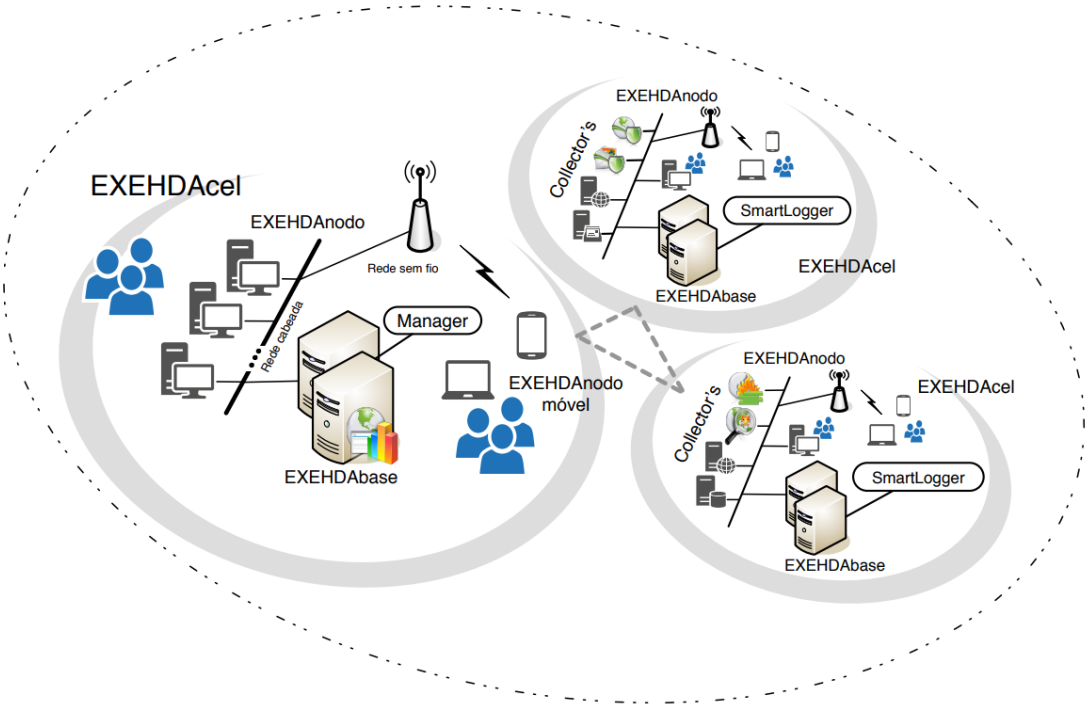

Figura 4: EXEHDA: Células de execução do ambiente ubíquo e seus componentes EXEHDA

Como referido anteriormente, o EXEHDA-USM (ALMEIDA, 2016) baseia-se nos conceitos do EXEHDA para efetuar o gerenciamento de segurança em uma rede de computadores, pois este *middleware* possui uma arquitetura distribuída que oferece suporte à aquisição, processamento e armazenamento de informações contextuais, além dos procedimentos de atuação sobre o meio, sendo estes fatores relevantes para a obtenção de Consciência de Situação.

Um ponto de vista importante do EXEHDA-USM pode ser analisado observando os seus componentes arquiteturais principais: Colector, SmartLogger e Manager. Mesmo visando finalidades diferentes ao longo do ambiente ubíquo, em cada um destes componentes é possível perceber fases de CS considerandos seus níveis de abstração. Opta-se por este enfoque para a discussão inicial dos componentes destacados:

- Percepção: nesta fase os eventos são coletados dos diversos sensores existentes no ambiente ubíquo. Envolve os processos de detecção, reconhecimento e monitoramento, que levam a consciência de múltiplos elementos situacionais, tais como, alertas relatados por sistemas de detecção e prevenção de intrusão, eventos registrados em logs, relatórios de análises de vulnerabilidades, bem como os seus estados atuais (tempo em que ocorreram, locais, condições, formas e ações) (ALMEIDA, 2016). No EXEHDA-USM as percepções iniciais

do ambiente ocorrem predominantemente no componente Colector mas pode também ser realizada diretamente no Smartlogger.

- Compreensão: síntese e correlação dos elementos desconexos identificados no nível de percepção por intermédio de diferentes estratégias, tais como, baseada em conhecimento e baseada em anomalias. O Smartlogger e o Manager detém papel fundamental neste processo, por já contarem com dados de eventos suficientes para efetuarem o processamento e análise das situações identificadas. A distinção entre estes dois componentes, em se tratando da fase em questão, está principalmente na abrangência dos dados coletados. Enquanto o SmartLogger trabalha somente no âmbito da célula da qual pertence, o Manager tem a disposição uma extensão maior de dados oriundos de várias células e da própria projeção dos SmartLogger's instalados no ambiente. O foco central desta fase é a integração destas informações no intuito de descobrir e avaliar o impacto que determinados eventos combinados podem ter no parque computacional como um todo ou em determinado seguimento.

- Projeção: a projeção é alcançada por meio do conhecimento da situação, da dinâmica dos elementos contextuais e das compreensão da situações de risco detectadas. Considerando estes aspectos a fase de projeção irá atuar no ambiente afim de minimizar ou precaver a possibilidade de nova ocorrência da situação e/ou diminuir o impacto ao ambiente. De forma geral, por contar com uma visão maior do ambiente, a atuação com este intuito é realizada pelo SmartLogger ou Manager no EXEHDA-USM.

- Comunicação: a comunicação entre os componentes respeita a hierarquia previamente definida. Desta forma, cada componente conta com os dados do contexto celular ao qual está inserido. Por fim, as informações tratadas são enviadas para o nível superior de análise e compreensão, de forma geral um Manager, que terá visão mais abrangente do ambiente como referido anteriormente.

A figura 5 mostra o funcionamento das fases de CS no SmartLogger, que é o componente no qual este trabalho tem maior foco.

O estágio de compreensão oportunizou a iniciativa do presente trabalho pois suas funcionalidades alinham-se com o uso de ontologias. Contando com os dados disponíveis em um SmartLogger, por exemplo, é possível a inferência de diversas situações por intermédio de regras e também pela observação do conhecimento que a ontologia é capaz de prover.

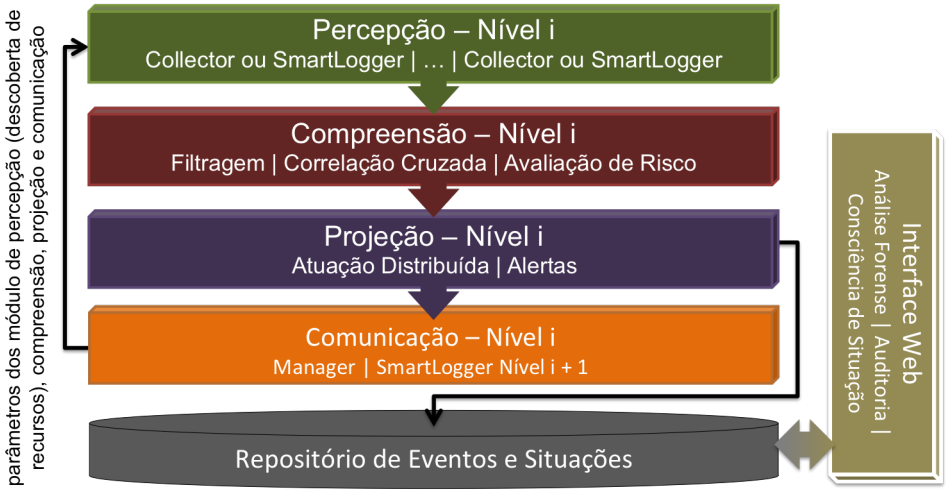

Figura 5: Smartlogger: CS no SmartLogger

3.2 Ontologias

Uma parte significativa do ecossistema tecnológico de aplicações baseadas em ontologias pode ser visualizada na adaptação do mapa conceitual, apresentada neste trabalho e exposta na figura 6 (ISOTAMI SEIJI ; BITTENCOURT, 2015). Contando com esta representação serão visitados diversos conceitos e soluções relevantes para a área de ontologias, como (i) quais são os formatos disponíveis para a representação de conhecimento de forma gráfica e formal e as implicações inerentes ao seus usos; (ii) os repositórios existentes e suas potencialidades; (iii) os diferentes *reasoner's* que podem ser utilizados para realização de inferências e (iv) outras soluções de ordem tecnológica que complementaram o escopo deste trabalho. Os nós destacados em tom mais escuro na figura 6 são as soluções que foram exploradas com maior ênfase na EXEHDA-SO, tendo então sua discussão ampliada.

3.2.1 Editores

Os editores de ontologia auxiliam muito não apenas na escrita e definição do funcionamento da ontologia, mas também em sua documentação. Um exemplo é o *Hozo*, editor gráfico para ontologias oriundo do Japão e que está atualmente em sua versão 5.2.30. A linguagem nativa do Hozo é XML *eXtensible Markup Language* e ele é capaz de exportar as ontologias utilizando as principais linguagens de representação utilizadas atualmente (KUMAZAWA et al., 2009). Outro editor que vem sendo aderido pela comunidade é o Fluent Editor[1], que conta com uma série de funcionalidades para escrita da ontologia em linguagem natural. Fluent importa e exporta em OWL[2] (*(Web Ontology Language)*), trabalha com regras SWRL (*Semantic Web Rule Language*) e consultas SparQL (SPARQL *Protocol and RDF Query Language*) que são soluções

[1]Cognitum Cognitive Platform - http://www.cognitum.eu/semantics/FluentEditor/
[2]Web Ontology Language - https://www.w3.org/OWL/

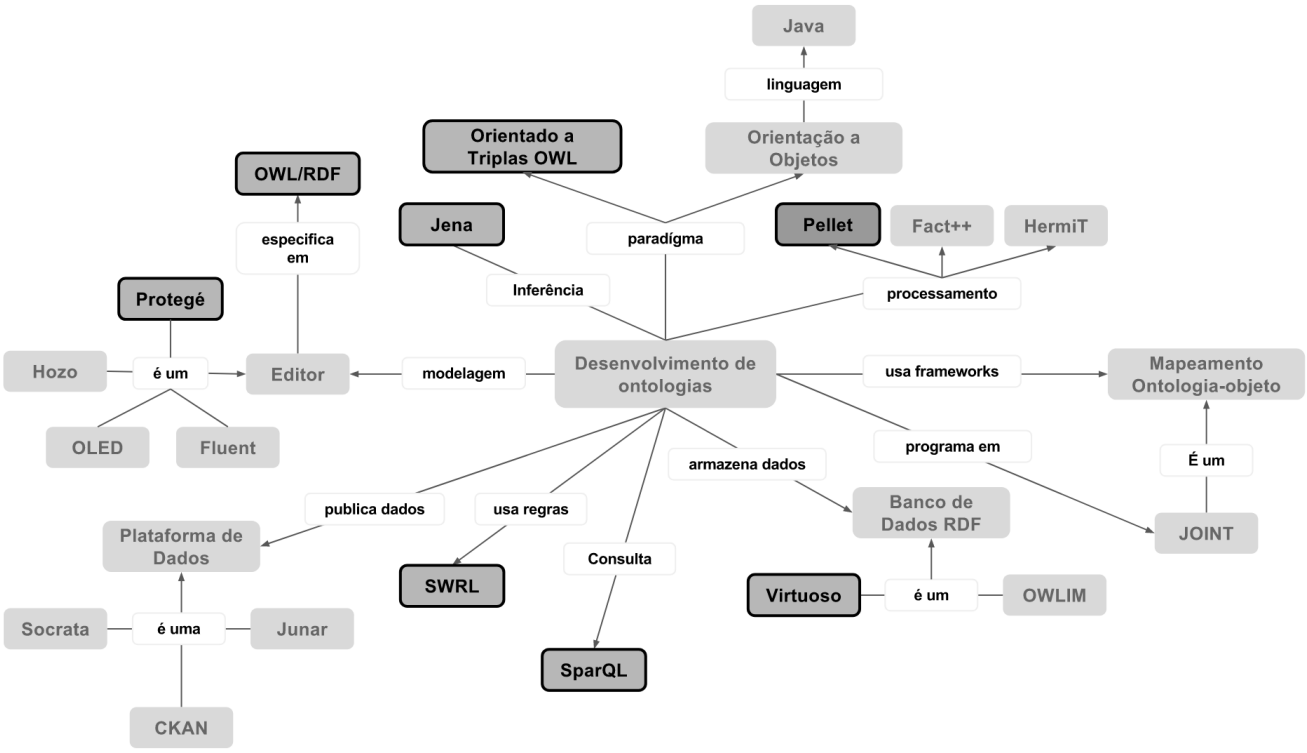

Figura 6: Mapa conceitual sobre o desenvolvimento de aplicações baseadas em ontologias. Fonte: Adaptado de (ISOTAMI SEIJI ; BITTENCOURT, 2015)

importantes para o presente trabalho e que são discutidas ainda neste Capítulo.

Já o OLED[3] *(OntoUML lightweight editor)* é um ambiente para o desenvolvimento, avaliação e implementação de ontologias de domínio em um modelo de linguagem OntoUML, o qual baseia-se na UFO *(Unified Foundational Ontology)*. O OLED fornece um conjunto simples, leve e integrado dos recursos, tais como a verificação sintática, simulação visual, inferência, validação das relações de paternidade, etc. Por fim tem-se o editor Protégé, *Open source*, conta com grande adesão e apoio da comunidade acadêmica sendo escolhido para o desenvolvimento deste trabalho.

Além da versão *desktop*, atualmente em sua versão 5,2.0, o Protégé também conta com uma versão web, utilizando Tomcat e MongoDB. Protegé dispõe de diversas funcionalidades, além da simples definição de classes e suas relações, voltadas para o gerenciamento do conhecimento nelas instanciadas. Por intermédio das propriedades de objeto e de dados permite estabelecer classes e seus possíveis valores para testes. Permite ainda a definição de instâncias bem como a edição de regras com o auxílio do plugin denominado SWRLTab.

[3]OLED - https://code.google.com/p/ontouml-lightweight-editor/

3.2.2 Repositórios

Impulsionada pelas demandas de *Big Data*, a utilização de bancos de dados não-relacionais vem ganhando espaço gradativamente. *NoSQL*, traduzido pela comunidade como *'Not only SQL'*, refere-se a um grupo cada vez mais familiar de sistemas de bancos de dados não-relacionais, nos quais a base de dados não é constituída de tabelas/esquemas e geralmente não são utilizadas funções em SQL *(Structured Query Language)* para manipulação de dados. Estas soluções são utilizadas em aplicações que trabalham com enormes quantidades de dados e, também, quando não é possível representar a natureza dos dados no modelo relacional de banco de dados (MONIRUZZAMAN; HOSSAIN, 2013). NoSQL destaca-se também por apresentar capacidade de distribuição da solução de banco de dados como um todo, e trabalhar de forma eficiente em textitcluster's (SADALAGE; FOWLER, 2013). No EXEHDA-USM, por exemplo, é utilizado o MongoDB para eventos, que é um banco de dados *NoSQL* orientado a documentos para armazenamento de dados conceituais.

No caso do presente trabalho, como repositório dos dados da ontologia desenvolvida foi escolhido o Virtuoso, que consiste em um servidor multiprotocolo que provê acesso ao banco de dados por meio de drivers ODBC/JDBC. Possui um motor de busca SQL e um servidor HTTP *(Hypertext Transfer Protocol)* para usuários administradores, com terminais em diferentes protocolos (ex. serviços web) e linguagem de *script* interna (ISOTAMI SEIJI ; BITTENCOURT, 2015).

O Virtuoso também oferece um motor de busca em SPARQL, que "traduz" as consultas em SPARQL feitas pelo desenvolvedor para a correspondente em SQL. SPARQL é também uma recomendação da W3C[4], integrando o projeto EXEHDA-SO. As consultas SparQL facilitam o processo de pesquisa de situações de segurança nas instâncias da ontologia permitindo consultas dos tipos *Select, ASK, Construct, Describe, Insert e Delete*.

3.2.3 Representação do Conhecimento

Devido as diversas possibilidades existentes na interlocução cotidiana, independente da lingua utilizada, a escolha do conjunto de termos que melhor represente um determinado conhecimento é uma das questões centrais para o desenvolvimento de ontologias. O vocabulário deve estabelecer uma identificação objetiva, livre de ambiguidades e duplicidades, afim de que o conhecimento possa ser processado e gere corretas inferências. Outro fator que corrobora com a adequada interpretação de dados em ontologias é a definição de restrições, estipulando relações lógicas existentes entre as classes desenvolvidas. Observa-se aqui que as restrições devem ser definidas de forma a não inviabilizar contextos externos ao ambiente em questão pois

[4] É um consórcio internacional formado por organizações, uma equipe em tempo integral e o público, que trabalha para desenvolver padrões para a Web. http://www.w3.org/

hierarquias verdadeiras em uma determinada organização podem não existir em outra.

Uma das maneiras mais usuais para representação do conhecimento em forma gráfica é o UML[5] *(Unified Modeling Language)*. A visualização das classes e relações existentes em determinado contexto facilita o entendimento do domínio proposto eximindo a análise de, por vezes extensas, linhas de código. Em se tratando de representação formal de ontologias, voltadas para a interpretação de computadores, as linguagens mais populares são RDF[6]/RDF-S[7] e OWL. RDF é uma especificação proposta pela W3C (*World Wide Web Consortium*) que permite indicar relações entre dados por intermédio de triplas, constituidas pelos nós denominados sujeito e objeto, e uma ligação mostrando a relação entre estes nós denominada predicado. As possibilidades trazidas pelo uso destas triplas, que também são referidas na literatura como vocabulários RDF, podem dar uma amostragem geral de determinado conhecimento, mas para que se avance em riqueza semântica destas triplas, podem ser definidas tags com a extensão RDF-S. Com RDF-S é possível então estabelecer classes, propriedades e relacionamentos. E voltada para aplicações que requeiram processamento deste conhecimento, extendendo a expressividade de RDF, disponibilizando outros elementos e seguindo sintaxe similar, temos a OWL.

Também recomendada pela W3C, OWL é uma linguagem declarativa para ontologias que, como referido anteriormente, mantém a característica de representação de conhecimento no formato de triplas, bem como o Esquemas estabelecido em RDF, extendendo seu vocabulário, auxiliando a interpretação e processamento do conhecimento por computadores. Para efetuar a modelagem do conhecimento, OWL utiliza Entidades para referenciar algum conceito ou item do mundo real. Por exemplo, pode-se representar o grupo de todos os ataques em redes de computadores pela entidade ≪attacks≫, identificada por uma IRI *(Internationalized Resource Identifier)* correspondente. Já combinações de diversas entidades podem ser consideradas com o uso de expressões, gerando novas entidades e derivando assim descrições mais completas sobre determinado item. E com o uso dos chamados Axiomas pode-se realizar inferência sobre as entidades consideradas. Ao considerar-se *bruteforce* uma *SubClassOf* de ”attacks”entende-se, por exemplo, que uma determinada instância de *bruteforce* é também uma instância de *attacks*. Outra método utilizado para inferir conhecimento sobre os dados instanciados na ontologia consiste na aplicação de regras. SWRL, baseada nas sublinguagens OWL DL (Ontology Web Language Description Logics) e OWL lite, é a linguagem voltada para este fim, extendendo as funcionalidades do conjunto de axiomas possíveis em OWL. Regras em SWRL permitem diversas

[5]Disponível em: ¡http://www.uml.org¿
[6]Resource Description Framework - https://www.w3.org/RDF/
[7]RDF Schema - https://www.w3.org/TR/rdf-schema/

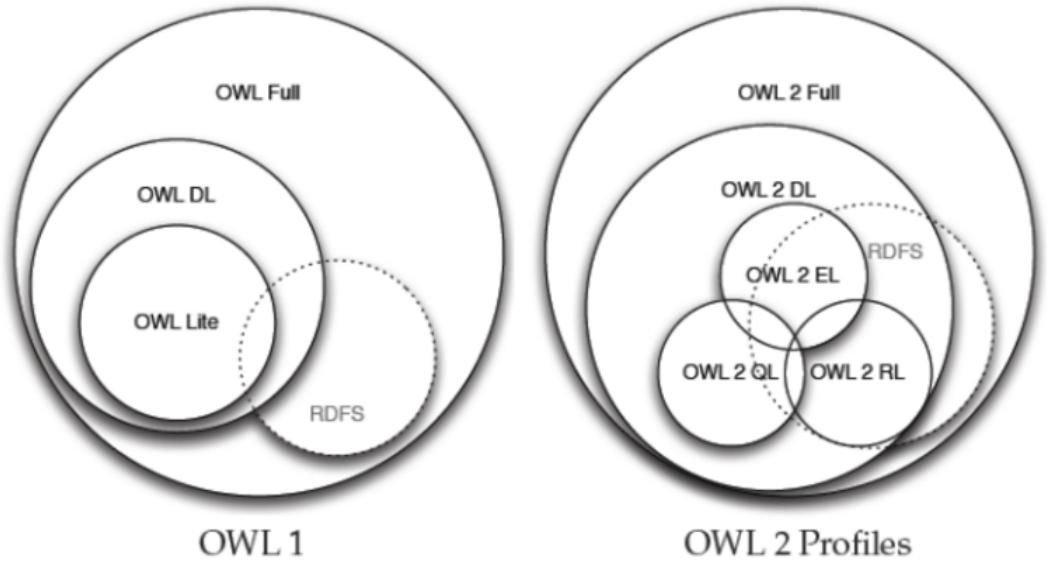

Figura 7: Níveis de expressividade da linguagem OWL. Fonte: (ISOTAMI SEIJI ; BITTENCOURT, 2015)

análises objetivas dos dados instanciados, como comparações incluindo as opções *equal, notEqual, lessThan, lessThanOrEqual, greaterThan* e *greaterThanOrEqual*. Há ainda construtores voltados para análise numérica, de *strings*, de tempo e verificação booleana.

A figura 7 ilustra os níveis de expressividade das versões 1 e 2 de OWL. A primeira versão ainda contava com a opção OWL Lite, que era uma opção limitada de OWL DL. Já na versão corrente inclui-se a nomenclatura perfis de OWL em atenção aos focos diferentes de cada fragmento da OWL. O perfil OWL Full é tido como o mais abrangente de todos, proporcionando maior liberdade sintática por intermédio de grafos RDF. Já OWL DL tem foco na chamada semântica direta, por intermédio de seu estilo baseado em lógica de descrição, sendo uma versão mais restrita e simplificada do perfil OWL Full sem perder a decidibilidade computacional. O perfil OWL EL, por sua vez, contempla apenas a família EL da lógica de descrição. O perfil OWL QL (*Web Ontology Language Query Language*) é voltado para a utilização com SGBDR (Sistemas Gerenciadores de Bancos de Dados Relacionais) tradicionais. E, por fim, tem-se OWL RL(*Web Ontology Language Rule Language*), suportando semântica baseada em RDF ou em lógica descritiva, este perfil preocupa-se com a questão de escalabilidade de raciocínio.

Em se tratando da serialização dos documentos em OWL é seguida estrutura sintática baseada em RDF/XML, que pode ser reconhecida por qualquer software OWL. Existem ainda outros formatos para serialização, conforme pode-se observar na tabela 1. O documentos criados no formato OWL/XML tem o intuito de processar documentos OWL por intermédio de ferramentas XML. A sintaxe funcional, por

sua vez, trás uma maneira simples de visualizar e compreender documentos OWL, enquanto a sintaxe Manchester é voltada para manipulação de estruturas em lógicas de descrição. Por fim, tem-se a família de formatos de serialização Turtle, que tem um foco voltado para Bigdata. Conta com o formato N-Triple, sendo versão mais simplificada de definir triplas; Turtle em sí, que também apresenta leitura bem simples, adicionando a possibilidade de descrever prefixos e IRIs relativos na estrutura do documento. Há ainda a extensão do formato Turtle denominado TriG que tem as mesmas características de Turtle contemplando a representação de múltiplos grafos e N-Quads que é uma extensão de N-triples e que permite intercâmbio de catálogo de dados.

Tabela 1: Formatos de serialização OWL e seus propósitos

Serialização	Status	Propósito
RDF/XML	Mandatório	Reconhecido por qualquer software OWL
OWL/XML	Opcional	Processamento simples com ferramentas XML
Sintaxe Funcional	Opcional	Visualização simples da estrutura formal da OWL
Sintaxe Manchester	Opcional	Leitura/Escrita simples de Ontologias em LD
Turtle	Opcional	Leitura/Escrita simples de triplas RDF

3.2.4 Processamento

As inferências dentro da ontologia são efetuadas pelos chamados *reasoner's*, podendo ser por exemplo o Pellet, o Fact++ ou o Hermit. Estes *reasoner's* podem ser utilizados em conjunto com o Jena (JENA, 2014), também de livre uso, que é um *framework* que provê a construção de web semântica e aplicações de *Linked Data*, fornecendo uma abstração e interface para manipulação de grafos RDF representados na memória principal e apoiados por mecanismos de banco de dados. Jena suporta diversos sistemas de banco de dados como MySQL, Postgres, Oracle e Virtuoso. Os dados armazenados podem ser recuperados por intermédio de consultas SPARQL.

3.3 Tecnologias Complementares

Para além das tecnologias direcionadas ao desenvolvimento de ontologias, também são utilizadas linguagens e ferramentas voltadas as fases de percepção e projeção de eventos. Estas são destacadas nas sub-seções seguintes.

3.3.1 Python

Afim de facilitar questões de compatibilidade com a programação dos módulos do EXEHDA-USM, optou-se pela utilização da linguagem de programação Python no desenvolvimento deste trabalho. Trata-se de uma linguagem multiplataforma e de código

aberto. Atualmente na versão 3.6, o Python é reconhecido como uma linguagem de fácil implementação e manutenção contando com diversos módulos de terceiros e bibliotecas que auxiliam em muitas tarefas básicas.

3.3.2 Logstash

Logstash é um framework de código aberto para centralização de *logs* que disponibiliza *parsing* dos dados monitorados de uma rede, sejam eles estruturados ou não estruturados, de diversos formatos e volumes, aceitando inclusive entradas de múltiplas fontes de dados. Este agregador facilita a visualização dos eventos e possui uma série de *plugins* de *output* como por exemplo CSV (Comma-separated values), HTTP (*Hypertext Transfer Protocol*), MongoDB, *File*, etc. Tradicionalmente utilizado em conjunto com o *ElasticSearch* (servidor de buscas) e com o Kibana (interface gráfica) formando a pilha ELK (*Elasticsearch, Logstash e Kibana*). O *Logstash* é utilizado para instanciar a ontologia desenvolvida na prototipação dos estudos de caso deste trabalho. Neste trabalho o *Logstash* é utilizado na fase de percepção para o pré-processamento dos eventos coletados, normalizando e contextualizando os dados brutos.

3.3.3 Plataforma Beats

A plataforma Beats também integra-se à fase de percepção deste trabalho por possibilitar a filtragem e compreensão de eventos, bem como comunicação encriptada dos dados, tolerância a falhas e balanceamento.

3.4 Considerações Finais

Este Capítulo apresentou uma visão geral sobre a base tecnológica na qual o trabalho foi desenvolvido. Primeiramente foram verificados alguns conceitos gerais do *middleware* EXEHDA de forma a introduzir o EXEHDA-USM, abordagem para gerenciamento de segurança em ambientes ubíquos desenvolvido. No trabalho de Almeida (2016) , Observou-se que a fase de compreensão do EXEHDA-USM oportunizava a atuação ontológica, contribuição na qual este trabalho concentrou seu esforço. As soluções e métodos utilizados na prototipação deste projeto são de forma geral tecnologias livres e soluções já consolidadas.

4 TRABALHOS RELACIONADOS

A busca por ciência de situação (CS) pode ocorrer valendo-se de diversas vertentes e fontes. No domínio de segurança da informação (SI), foco deste trabalho, estas fontes podem ser caracterizadas por registros derivados desde equipamentos de rede como *switchs* e roteadores, por logs de serviços ou aplicações, por ferramentas especificas de análise de tráfego, entre outras. Bem como constata-se heterogeneidade quanto a origem dos dados contextuais, também verifica-se o mesmo em se tratando das abordagens para processamento destes dados.

Pontuando que a etapa de compreensão de CS é uma das principais verticais deste trabalho, a amostragem de trabalhos relacionados apresenta, nas próximas seções, a aplicabilidade das abordagens ontológicas no provimento de bases para compartilhamento de conhecimento e no processamento de eventos refletindo as situações de interesse. Serão destacados trabalhos que utilizam ontologias visando proporcionar contexto, limitando o escopo do levantamento a trabalhos que tenham a SI como artifício ou finalidade.

4.1 Interoperabilidade de Contextos

Uma das características da UbiComp recai sobre a chamada interoperabilidade, isto é, capacidade de cooperação entre protocolos e serviços independentes. No domínio de SI a interoperabilidade representa um desafio presente por conta da variedade de serviços e dispositivos que dão suporte as redes de computadores, como mencionado no Capítulo 2. Considerar informações destes diversos ativos, que muitas vezes não apresentam sequer representação compatível, caracteriza o aproveitamento de contextos independentes para gerar inferências. A centralização do entendimento e unificação do formato, em resposta a demanda de análise de eventos oriundos de fontes distintas, é uma das vantagens das abordagens ontológicas.

O trabalho denominado *The Evaluation Process of a Computer Security Incident Ontology* (MARTIMIANO; MOREIRA, 2006) faz uso de ontologias dentro do domínio de SI, onde a denominada OntoSec objetiva o estabelecimento de uma estrutura

única para representar informações sobre incidentes de segurança, possibilitando a correlação dos eventos percebidos.

A figura 8 ilustra os conceitos de mais alto nível do domínio de incidentes de segurança e como eles se relacionam, retratando o primeiro de quatro níveis de classes da OntoSec. Em uma observação geral se pode verificar que este *Core* contempla, por exemplo: um agente (*agent*) que possa efetuar um ataque (*attack*) causando um possível incidente de segurança (*security incident*). Para executar um ataque, um agente pode usar uma ferramenta (*tool*), a qual pode explorar uma vulnerabilidade (*vulnerability*) para obter determinado acesso. Um incidente de segurança implica em consequências (*consequence*) e ações nos ativos (*assets*). Um incidente de segurança também pode ter pré-condições (*pre conditions*), e estas pré-condições podem estar relacionadas à vulnerabilidades denotando condições específicas nas quais determinado incidente é possível. A consequência pode estar relacionada a um ativo e um incidente pode preceder ou suceder outro incidente. Uma vulnerabilidade pode ser explorada por outras vulnerabilidades precedentes ou posteriores. Uma vulnerabilidade pode ainda ter uma correção (*correction*), a qual é desenvolvida por um fornecedor (*supplier*) da organização.

A OntoSec deriva diversos de seus conceitos do Projeto CVE *(Common Vulnerabilities and Exposures)*, que é um modelo de referência público de vulnerabilidades conhecidas mantido pelo MITRE[1]. Como fonte de eventos da ontologia são mapeadas automaticamente as detecções originalmente percebidas pelo Snort[2]. Uma vez mapeados os eventos na ontologia, são possíveis consultas na direção de diversos aspectos de interesse, como por exemplo:

- Quantos incidentes ocorreram por conta de uma determinada vulnerabilidade?
- Quais foram os tipos mais comuns de incidentes?
- Quais foram os ativos que apresentaram mais problemas de segurança e merecem maior atenção e/ou atualizações?
- Quais foram as portas de comunicação mais utilizadas para a realização de ataques?

Estas respostas são obtidas por intermédio de combinações de consultas SparQL sobre as instâncias das detecções do Snort. O trabalho conclui que o uso de ontologias de incidentes de segurança nas organizações pode melhorar sua capacidade de gerenciar e controlar problemas, otimizando a tarefa de gerência de rede e permitindo análises mais assertivas. O trabalho não explora os conceitos de ciência de contexto,

[1] MITRE - corporação americana sem fins lucrativos que coordena diversos centros de pesquisa do governo federal

[2] Snort - sistema *open-source* para detecção de intrusão em redes

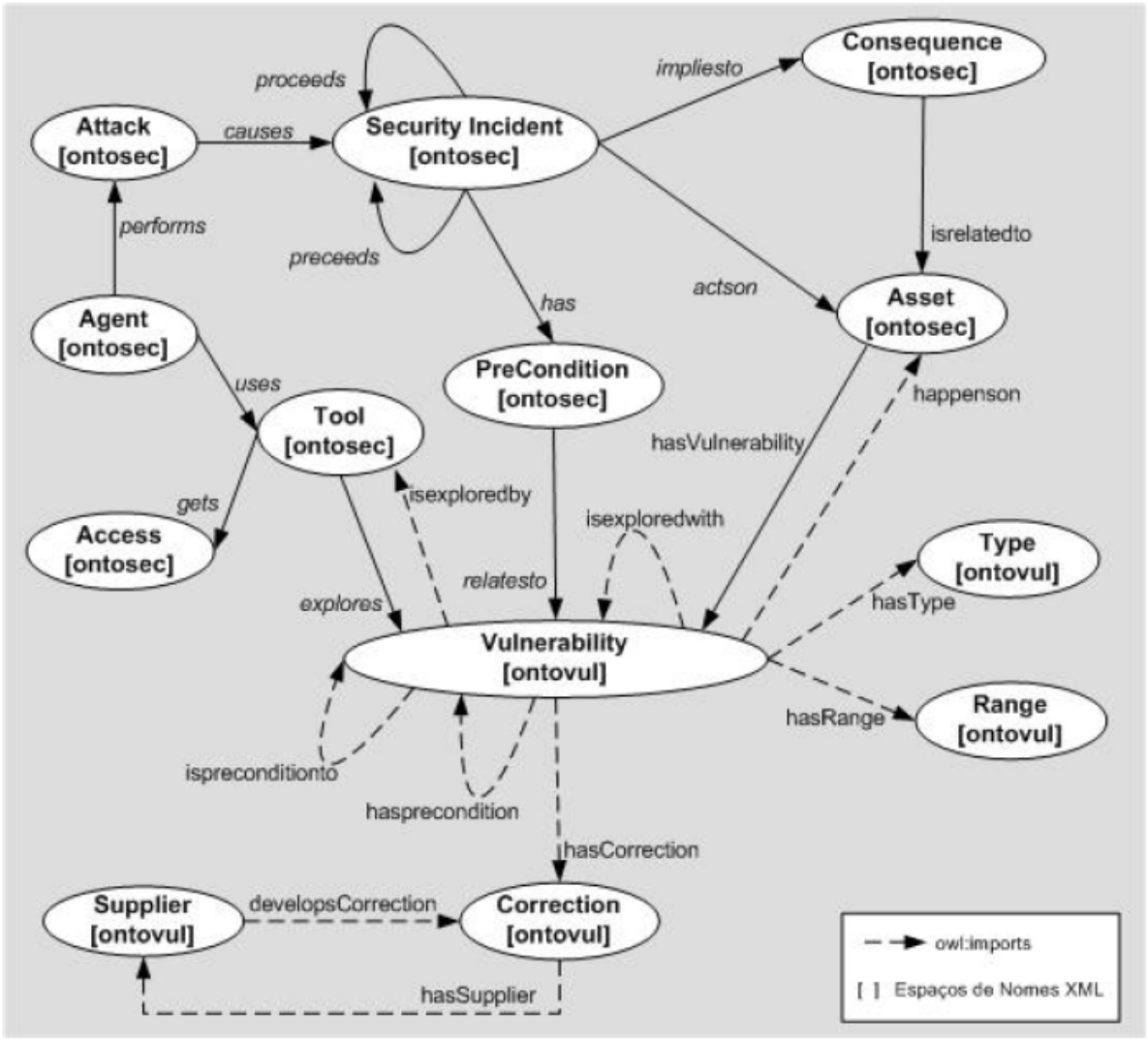

Figura 8: Principais conceitos relacionados ao OntoSec
Fonte: MARTIMIANO; MOREIRA, 2006

apesar de haver uma evidente correlação entre a solução desenvolvida e os referidos conceitos.

4.2 Base de Conhecimento para Arquiteturas

Minimizar a complexidade do gerenciamento de infraestruturas computacionais é um dos objetivos principais no uso de ontologias no domínio de SI. Ao formalizar as informações sobre o ambiente computacional, que por vezes não são devidamente documentadas, as ontologias auxiliam na busca por situações de risco. Neste sentido, o trabalho *An ontology of security applied to the business process of management* (AZEVEDO et al., 2009) apresenta um sistema autonômico baseado em ontologias e agentes Inteligentes para uso em SI, tendo como intuito resguardar os recursos de TI de agentes maliciosos. Para tal, como objetivos específicos os autores destacam: (i) o desenvolvimento de uma ontologia para representação de informações sobre SI especificando, tratando e mitigando riscos de segurança em diversos ambientes corporativos; (ii) o desenvolvimento de uma arquitetura autonômica baseada em ontologias com intuito de auxiliar os responsáveis pela segurança da informação na proteção, cura, otimização e configuração de sistemas computacionais corporativos. A arquitetura tem a capacidade de determinar o diagnóstico mais próximo da situação atual aprendendo autonomamente um novo diagnóstico para casos sem semelhanças nas ontologias e capacidade de considerar contexto ao prover soluções.

A ontologia utilizada neste trabalho é a CoreSec que foi apresentada pelo mesmo autor no trabalho (AZEVEDO et al., 2008). A CoreSec objetiva servir como base de conhecimento para apoio a decisões estratégicas dos responsáveis pela segurança computacional corporativa. A CoreSec é composta por 86 conceitos representados por classes, 39 propriedades do tipo *object property* e 31 do tipo *data properties*. Na figura 9, apesar de ser apresentado somente o núcleo da ontologia proposta, é possível perceber grande parte do teor da abordagem a partir das classes destacadas: vulnerabilidade, ativo, ferramenta, ataque, ameaça, risco entre outras. Uma breve descrição das principais classes é relacionada a seguir.

- Vulnerabilidade: possui instâncias relativas a vulnerabilidades que um determinado recurso pode ter. A partir desta classe partem diversas correlações como (i) *HasCorrection*, identificando possíveis correlações; (ii) *Permits*, indicando que a vulnerabilidade permite a ocorrência de um incidente; (iii) *IsExploitedby-toll*, identificando quais ferramentas utilizadas pelos agentes mal-intencionados exploram vulnerabilidades; (iV) *Happenson-Asset*, onde são descritas quais vulnerabilidades os ativos tem.

- Ativo: representa os ativos a serem protegidos de ataques e de incidentes de

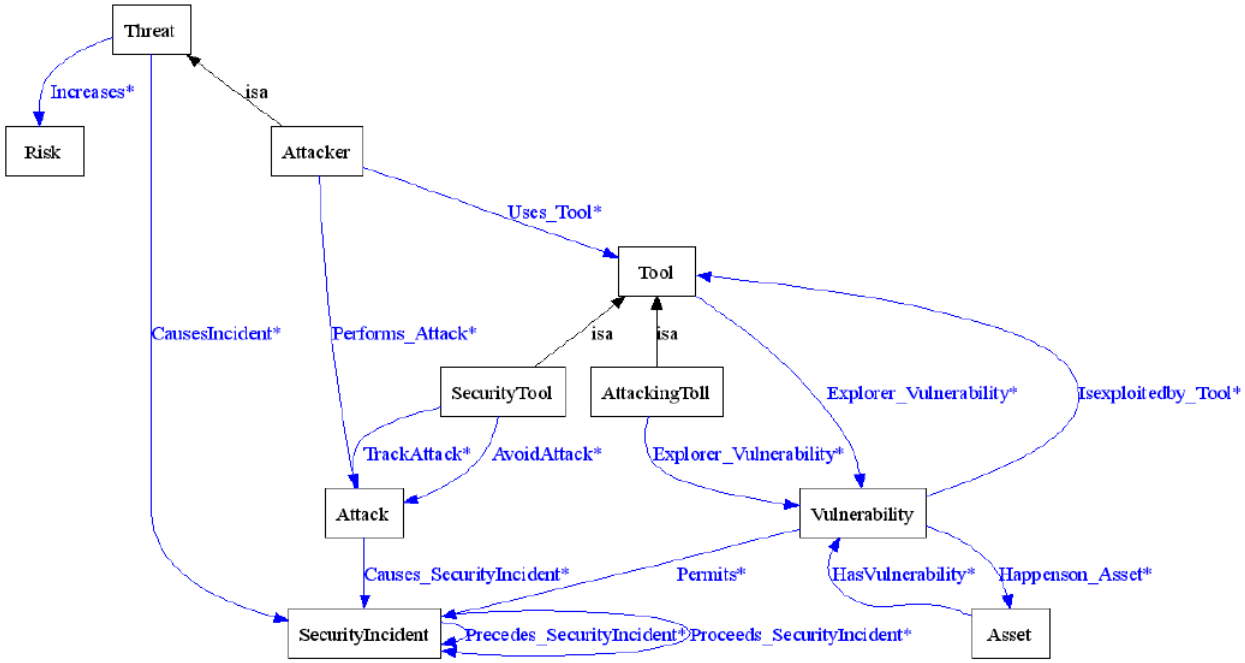

Figura 9: Classes e relacionamentos que formam o núcleo do CoreSec
Fonte: AZEVEDO et al., 2008

segurança. Possui a relação *HasVulnerability* indicando quais vulnerabilidades o ativo pode ter. A classe Ativo possui as subclasses (i) *hardware*, para equipamentos como *switches*, computadores pessoais, servidores, etc.; (ii) software's, para aplicações diversas; (iii) usuário, que pode representar tanto uma mera vítima final de algum incidente ou também o promotor do ataque em si e (iv) informação a respeito dos dados afetados.

- Ferramenta: possui instâncias que identificam ferramentas e mecanismos utilizados para explorar vulnerabilidades de determinado recurso, seja ele um usuário final, hardware ou software. A classe Ferramenta possui também subclasses diferenciando ferramentas maliciosas (malware, vírus) de ferramentas de segurança (Firewall, etc).

- Incidente de Segurança: esta classe possui as relações: (i)*HasConsequence*, indicando as consequências que um incidente pode causar para a organização; (ii)*HasImpact* para mensurar o impacto do incidente; (iii)*affects*, indicando que o incidente afeta os negócios da organização; (iv)*HasToleranceLevel*, indicando que a organização possui um nível de tolerância quanto ao impacto causado pelo incidente de segurança e (v)*HasCouterMeasure*, para possibilidades de mitigar o incidente.

- Ameaça: conta com as relações (i)*CausesIncident*, indicando uma ameaça causada por algum incidente de segurança; (ii)*Increase*, indicando que pode haver

um aumento nos riscos de segurança. Uma ameaça pode ser natural, ou causada por algum agente malicioso.

- Ataque: possui instâncias para os ataques realizados aos ativos possuindo os relacionamentos (i)*CausesSecurityIncident*, indicando que um ataque causa um incidente.
- Atacante: contando com os relacionamentos (i)*Performs-attack*, indicando que um agente realiza um ataque e (ii)*Uses-tool*, indica a ferramenta utilizada, representada pela classe ”ferramenta”.
- Risco: instancia os níveis de risco aceitáveis.

A figura 10 mostra a participação da ontologia Coresec na arquitetura da AutoCore ao lado de duas outras ontologias de aplicação e ilustra o caráter multiagente do trabalho.

O trabalho conclui que atinge o objetivo em desenvolver uma ferramenta autonômica focada em SI para auxilio nas tomadas de decisão quanto à proteção do ambiente.

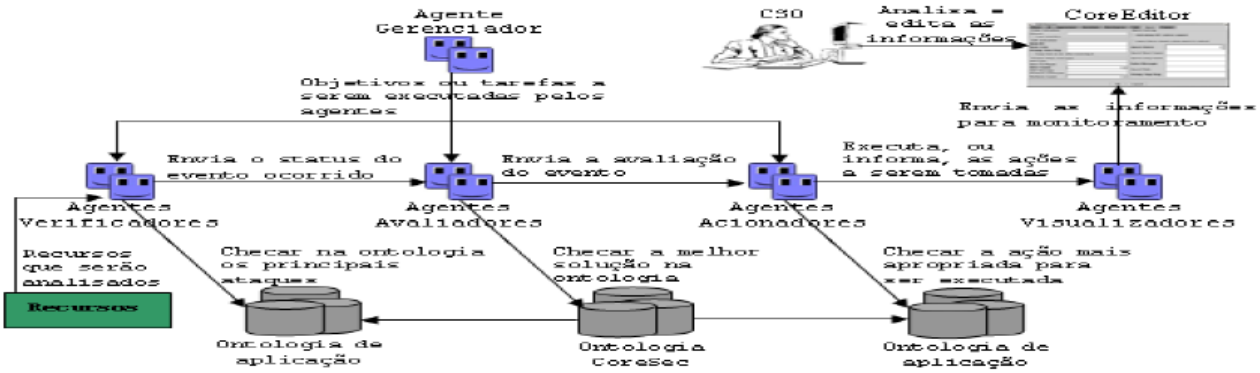

Figura 10: Arquitetura do AutoCore (AZEVEDO et al., 2008)
Fonte: AZEVEDO et al., 2008

4.3 Monitoramento de Segurança

Ao ponderar acerca do conceito de Tim Bass (BASS, 2000) para NSSA os autores do trabalho *Ontology based Approach for Perception of Network Security State* (BHANDARI; GUJRAL, 2014) pontuam proposta voltada para a etapa de percepção de situações, problematizando sobre a dinamicidade nas mudanças de *status* de segurança das redes e diversidade de ferramentas que, por vezes, geram uma quantidade de dados incompatível com a capacidade de análise dos profissionais envolvidos.

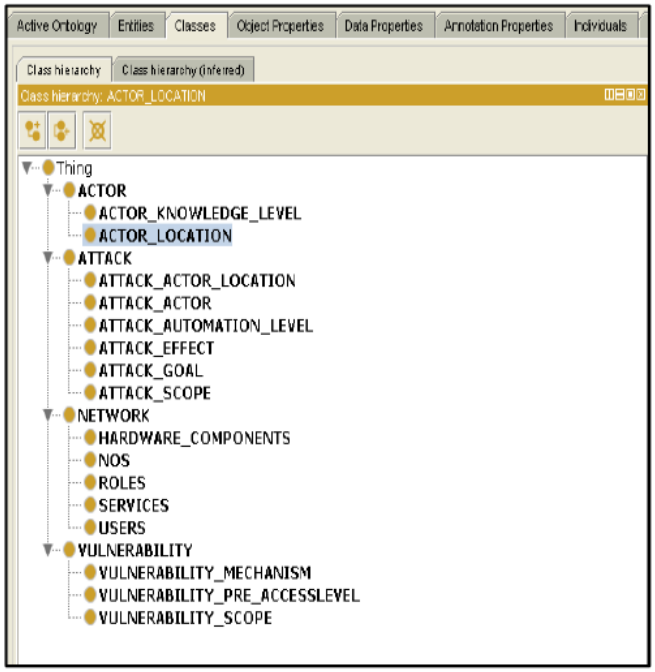

Figura 11: Classe hierárquicas da ontologia proposta (BHANDARI; GUJRAL, 2014)
Fonte: BHANDARI; GUJRAL, 2014

Primeiramente, a partir da classificação de segurança da informação (HANSMAN; HUNT, 2005) e da classificação de ataques (HEERDEN; LEENEN; IRWIN, 2013), no trabalho os autores estabelecem as classes da ontologia proposta, sendo que algumas classes originalmente sugeridas foram suplantadas por conta do escopo do trabalho. A classificação resultante, que pode ser vista na figura 11, conta com as classes: Ator, Ataque, Rede e Vulnerabilidade.

Ao receber como entrada as vulnerabilidades oriundas de uma ferramenta externa, a ontologia determina ou percebe o estado da rede, que pode ser “seguro”, “vulnerável”ou “inseguro”. Ainda são demonstradas as regras para prover esta indicação de situações de risco:

- se ”o ator é externo”e ”o nível de automação é automático”e ”o objetivo é destruir dados”e ”o escopo é a rede como um todo”e ”o valor do serviço afetado é maior que um dado limiar”então o estado da rede é insegura.
- se ”o ator é local”e ”o nível de automação é manual”e ”o objetivo é ler dados”e ”o escopo é uma pequena rede privada”e ”o valor do serviço afetado é menos que um dado limiar”então o estado da rede é moderadamente seguro.

O artigo (BHANDARI; GUJRAL, 2014) conclui que a predição do estado atual de uma rede, desenvolvida no trabalho, é um importante para uma adequada ciência situacional de segurança. Para o desenvolvimento da ontologia foram considerados conceitos estabelecidos pelas taxonomias *Common Vulnerability Enumeration* (CVE),

Common Weakness Enumeration (CWE), *Common Attack Pattern Enumeration and Classification*(CAPEC), caracterizando o reuso da proposta. O trabalho ainda define como promissora a abordagem envolvendo ontologias e regras, se mostrando uma alternativa escalável, flexível e de fácil adaptação a novos desafios envolvendo a segurança da organização.

4.4 Análise Posterior para Ambientes Críticos

A implementação de controles de segurança de certa maneira pode comprometer diretamente a disponibilidade em redes de comunicação ao efetuar contramedidas equivocadas à falso-positivos. Isto pode inclusive interferir no funcionamento de sistemas e na transparência das ações dos ambientes quando, por exemplo, o barramento de uma porta TCP (Transmission Control Protocol) ou UDP(User Datagram Protocol) é executado de maneira errônea e automática. Esta situação é tratada como ponto motivador do trabalho *Log content extraction engine based on ontology for the purpose of a posteriori access control*(AZKIA et al., 2014) o qual defende que quando uma redução nos pontos de checagem de segurança na rede se faz necessária, uma possível alternativa passa a ser a análise posterior dos acontecimentos registrados em arquivos de log. Neste cenário, anormalidades podem ser verificadas em um segundo momento para que não haja qualquer impacto sobre a operação do ambiente e para que as medidas sejam avaliadas com maior minúcia pelos analistas de redes. Esta medida pode ser entendida como um controle de segurança posterior e é defendida pelos autores para aplicações de assistência médica que precisam impreterivelmente de conectividade. A estratégia, apesar de válida para redução de pontos de checagem de tráfego e atender a criticidade de determinados ambientes, se mostra um desafio em alguns casos por conta da não periodicidade das checagens manuais, erros humanos na análise, etc.

O trabalho lida justamente com o domínio de *healthcare*, a partir de seus trabalhos relacionados pontua três elementos para que esta abordagem de controle posterior possa ocorrer:

- Processamento de Logs: grava o histórico das ações executadas.
- Análise de Logs (auditoria): detecta anormalidades nas ações por intermédio de verificação de logs com base em um conjunto de regras de segurança.
- Procedimento de Prestação de Contas: toma ações anormais como entrada e verifica se a disfunção foi autorizada.

Há um foco maior na etapa de Processamento de Logs, mais particularmente no modo como ocorre a extração do conteúdo dos logs. Em linhas gerais, o Processamento de Logs aceita como entrada qualquer formato de log para, na sequência,

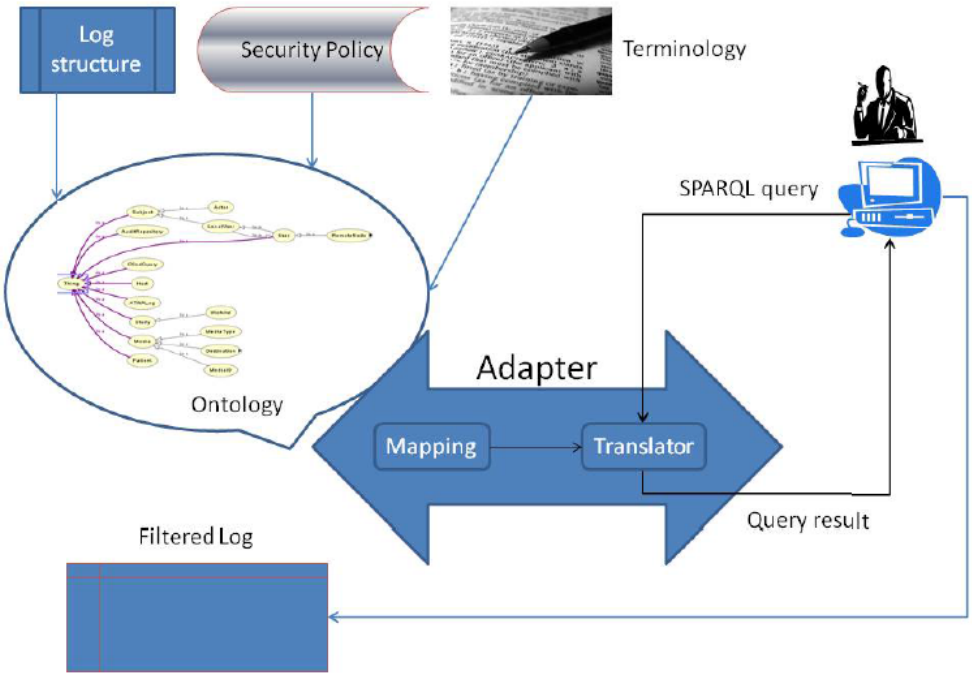

Figura 12: Modelo da engine de Logs
Fonte: AZKIA et al., 2014

reescrevê-los em um formato de representação de conhecimento comum. Esta transformação auxilia na identificação de possíveis incidentes ou anormalidades nos registros. Na sequência, o mecanismo de extração de conteúdo de logs, usando uma linguagem de pesquisa, pega como entrada os dados de registros no formato reescrito e gera como saída alguns dados úteis e essenciais. Os dados tratados nesse processo são escolhidos de maneira a serem confrontados com uma política de acesso que constitui uma segunda entrada da etapa de Processamento. A figura 12 transmite uma ideia deste processo, bem como mostra a atuação do *middleware Adapter* que tem por finalidade efetuar o mapeamento entre os dados de log armazenados e o modelo ontológico, bem como prover uma integração fácil para com a estrutura de logs por intermédio de consultas SPARQL.

A ontologia é composta basicamente por 3 elementos: (i) uma terminologia usada no domínio de *healthcare*; (ii) uma estrutura de logs oriundos dos sistemas; e (iii) uma política de segurança.

A terminologia de *Heathcare* mune a ontologia de classes referentes a conceitos da área médica como ”Relatórios de Consulta”, ”Dados de pacientes”, etc. Quanto à estrutura de logs, o estudo de caso trabalha com logs ATNA (*Audit Trail and Node authentication*) especificado pelo padrão IHE (Integrating the Health care Enterprise) como entrada. O referido log é focado no monitoramento de atividades ligadas a

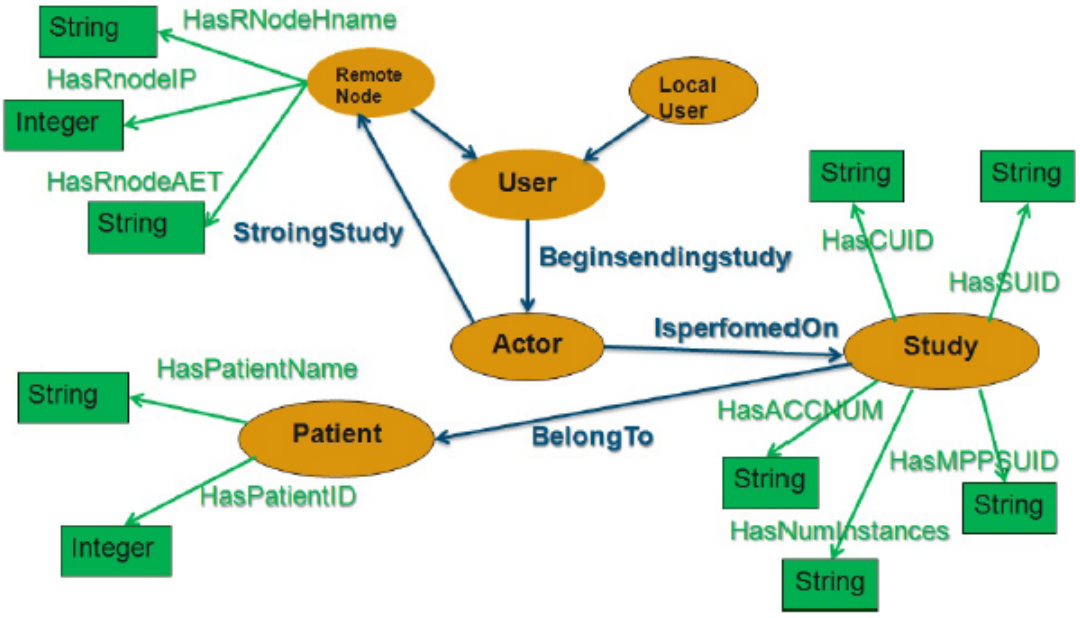

Figura 13: Ontologia BeginStoreInstances
Fonte: AZKIA et al., 2014

segurança, privacidade, autenticação e controle de acesso em aplicações distribuídas do domínio médico. Os registros são em XML e são compostos por:

- Host: define e especifica o hardware e software onde a ação ocorre. Pode ser um *actor (software ou hardware)* ou um *user* (usuário) para a classe *subject* da ontologia.
- TimeStamp: determina quando a ação ocorreu em um cenário auditado. (TimeStamp é um atributo essencial para determinar se um acesso foi dado ou não.).
- Auditable events: número fixo de eventos auditáveis com parâmetros relacionados.

O modelo de política de segurança, um terceiro elemento que compõe a ontologia, é adaptado de OrBAC (CUPPENS-BOULAHIA et al., 2008) e de RBCA (FERRAIOLO; KUHN, 1992), modelos genéricos para controle de acesso. O modelo resultante apresentado no artigo especifica a quádrupla: *subject; action; object; interval*. A Figura 13 mostra um dos eventos auditáveis mais importantes para o estudo: BeginStoreInstances.

São demonstradas ainda algumas consultas que incluem propósitos de monitoramento, verificações de disfunções e violações. Flexibilidade e adaptabilidade são dois dos benefícios destacados na conclusão deste artigo, oriundos da formalização e das consultas que a ontologia proporciona. A correlação de ações oriundas de logs

com políticas de segurança proporcionou boas perspectivas de detecção de possíveis violações e geram elevada consciência de situação ao ambiente mesmo que por intermédio de analise posterior.

4.5 Composição Híbrida de Processamento de Situações

Uma das funcionalidades mais exploradas em se tratando de ontologias no domínio de SI é sua capacidade de formalizar informações contextuais para gerar bases de conhecimento, que podem ser interpretadas por outras técnicas de processamento de eventos integrando um *framework* voltado para infraestruturas de larga escala. Um exemplo deste uso para ontologias é exposto no trabalho *A Generic Intrusion Detection and Diagnoser System Based on Complex Event Processing* (FICCO; ROMANO, 2011) onde é apresentado um sistema denominado *Intrusion Detection and Diagnois System* (ID^2 S). O ID^2 S tem foco no diagnóstico e na detecção de intrusão, implementando uma híbrida e hierárquica correlação de processos para a detecção de cenários complexos de intrusão. A capacidade de correlação de indícios de ataques é dirigida por uma base de conhecimento ontológica, capturando a relação causal entre as atividades maliciosas preliminares que são detectadas. Já a parte de consultas em tempo real de diferentes dados para detecção de cenários de ataque fica sob responsabilidade de Processamento de Eventos Complexos (Complex Event Processing).

A figura 14 ilustra a expressividade da ontologia na organização do conhecimento de SI, sendo possível perceber que esta é caracterizada por diversas entidades hierarquicamente organizadas voltadas para o Monitoramento de Ataques, Correlação, Detecção de Intrusão e Recuperação. A classe *attack* permeia diversos trechos desta hierarquia lógica, sendo que a partir do trecho *Attack Monitoring* pode ser descrita por *AttackIndicator* que possui predicados para (i) *hasTrustworthiness* referindo a probabilidade do evento detectado ser de fato um ataque, (ii) *isAssociatedTo* referindo a qual nível de verificação determinado evento será submetido as denominadas *probes*, (iii) *Indicates* ligando o indicador de ataque a um sintoma e por fim (iv) *isDirectedTo* ligando o ataque a um alvo. A classe *target*, em detecção de ataques, representa os tipos de ativos passíveis de sofrerem ataques.

A figura 15 mostra o framework ID^2 S de forma completa. Com as informações instanciadas na ontologia, o processamento dos cenários de ataques é efetuado por CEP, correlacionando as situações intermediárias encontradas. Uma vez detectados os cenários de ataque são então encaminhados alertas ao chamado agente remediador, vocábulo alusivo ao analista de sistemas. Os registros utilizados na prototipação são oriundos do IDS *Prelude*.

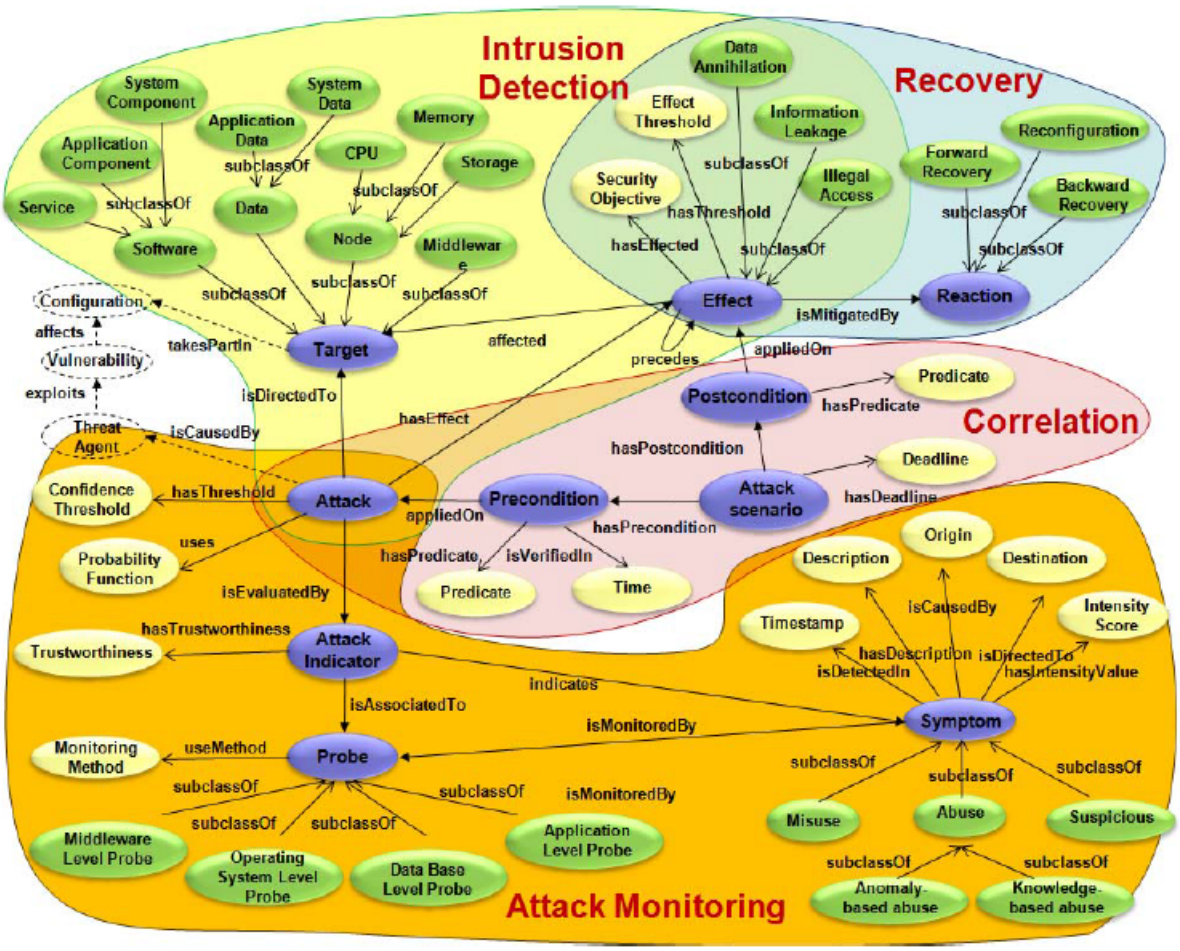

Figura 14: Visão simplificada da base de conhecimento
Fonte: FICCO; ROMANO, 2011

4.6 Considerações Finais

Os trabalhos relacionados que foram detalhados nesta seção exploraram o uso de ontologias como artifício na busca de CS, tendo como pressuposto o foco em SI. Percebe-se que as ontologias, neste domínio, tem por funcionalidade inferir dados distintos, relacionados ao projeto, ao domínio estabelecido e as tecnologias do entorno. Mais especificamente, as abordagens ontológicas demonstraram auxiliar nas análises de contextos independentes, isto é, correlações sobre dados de fontes distintas, que em ferramentas tradicionais de redes não são verificadas. Demonstrou-se ainda que bases de conhecimento formalizadas em ontologias podem complementar arquiteturas na busca por diversas situações de interesse, representando conceitos da SI ou do ambiente, otimizando a análise dos eventos detectados.

Percebe-se que, assim como ocorre em outras áreas, o reuso dos conceitos definidos pelas classes ocorre em muitos dos trabalhos analisados, o que vai com o passar do tempo especializando o método de ontologias para SI. Destaca-se que, salvo o

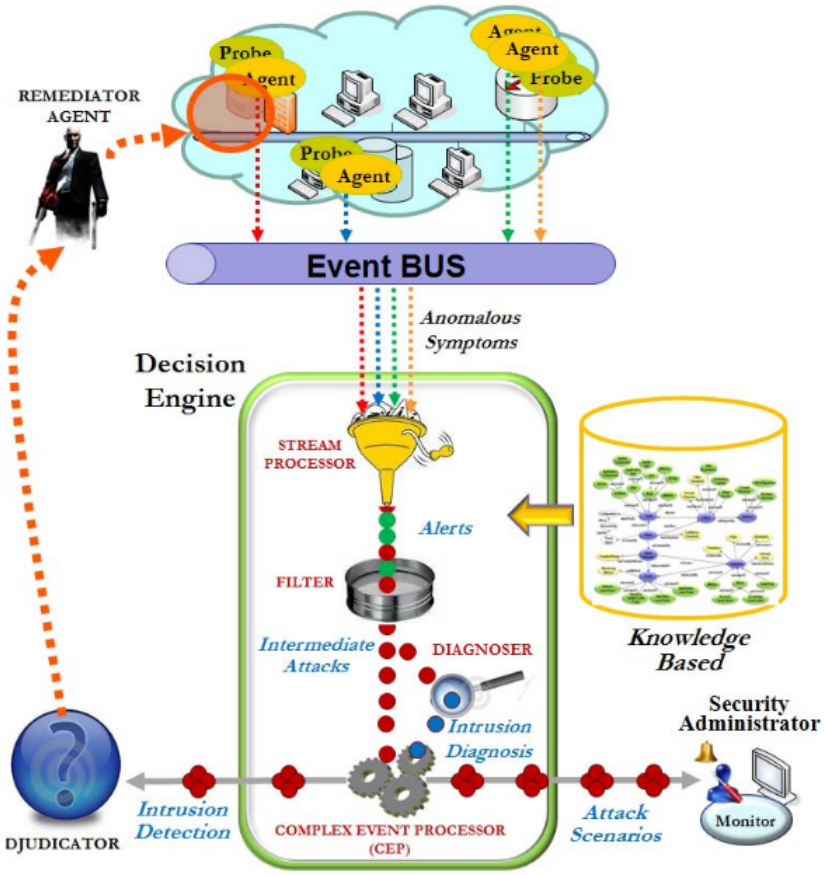

Figura 15: Framework ID^2 S
Fonte: FICCO; ROMANO, 2011

artigo (BHANDARI; GUJRAL, 2014), as abordagens relacionadas neste Capítulo não pautam suas estratégias sob a teoria conceitual de CS, apesar de possuirem relação.

Os trabalhos, em sua maioria, demonstraram seguir os padrões estabelecidos pela W3C ao utilizarem a formalização OWL ou RDF e consultas SparQL. A partir desta coleção de artigos foi possível obter uma visão abrangente do uso de ontologias em SI e a tabela 2 mostra uma comparação no que diz respeito as soluções utilizadas nas abordagens.

Tabela 2: Comparação entre os trabalhos ontológicos no domínio de SI

	Formalismo	Consultas	Inferências	Regras	Repositório	Eventos	Interação
MARTIMIANO; MOREIRA, 2006	OWL	SparQL	-	-	MySQL	SNORT	Java Jena
AZEVEDO et al. 2009	OWL	SparQL	Pellet	-	-	-	Java Jena
BHANDARI; GUJRAL, 2014	OWL	-	Hermit	SWRL	-	-	-
AZKIA et al., 2014	RDF/OWL	SparQL	-	-	-	IHE-ATNA	-
FICCO; ROMANO, 2011	-	-	-	CEP	-	Prelude	-

5 EXEHDA-SO

Os Capítulos 2 e 3 trouxeram, respectivamente, a base conceitual e as tecnologias que alicerçaram o desenvolvimento deste trabalho. Discutiu-se o alinhamento existente entre o uso de ontologias no processo de fortalecimento de linhas de defesa em Segurança da Informação (SI) por intermédio da decorrente Ciência de Situação (CS) que é proporcionada pela técnica. O modelo denominado EXEHDA-SO (*Execution Environment for Highly Distributed Applications - Security Ontology*), tema deste Capítulo, busca justamente alinhar as potencialidades de ontologias ao domínio de SI no intuito de buscar inferências em contextos específicos, que possam auxiliar a análise de ações maliciosas em ambientes ubíquos, adaptando-os de forma mais assertiva quando necessário.

São inúmeras as ferramentas e métodos de segurança que podem ser implantados em um ambiente computacional de larga escala para o fortalecimento da segurança. Estas soluções podem ser focadas em um aspecto específico do ambiente ou podem ser generalistas. Em um *firewall*, por exemplo, permite-se deliberar quais portas de comunicação podem ser ou não acessadas, em ambas redes interna e externa. Já em um *Web Application Firewall* (WAF) serviços web são monitorados em tempo real, provendo controle de acesso especificamente para este tipo de aplicação. Entretanto, a forma com a qual estas soluções são geridas ainda configura um desafio importante em SI, pois continuamente surgem novas técnicas, falhas e vulnerabilidades, a medida que são disponibilizados novos recursos tecnológicos, demandando constantes ajustes e adaptações no funcionamento e configurações vigentes.

Quando ações inadequadas são percebidas neste conjunto de soluções, são gerados registros que funcionam como sensores do estado do ambiente. A reação à estas ações pode determinar a extinção completa ou parcial do problema ou pode gerar indisponibilidades indesejadas. Então, interpretar o contexto no qual ocorrem estas ações configura também um desafio para a maximização de controles efetivos que não derivem novos problemas. Também sob este aspecto, a dificuldade nesta questão deriva principalmente da elevada diversidade e quantidade de eventos sensoriados continuamente.

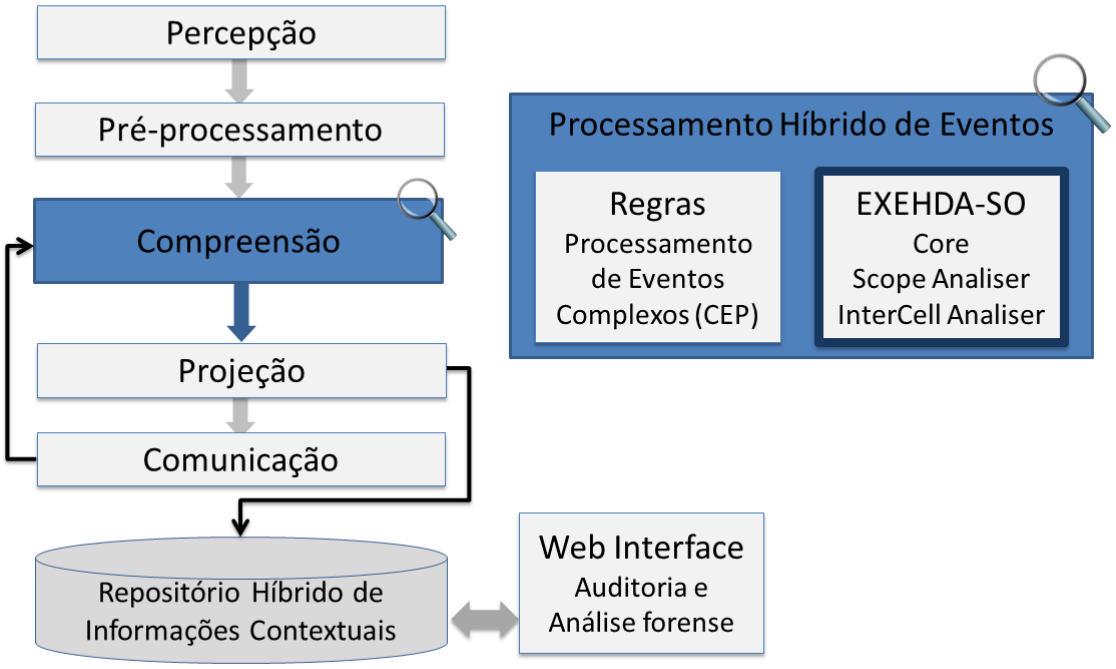

Figura 16: EXEHDA-SO integrando a etapa de Compreensão

Assim, pontua-se que a EXEHDA-SO objetiva a concepção de uma abordagem ontológica, sendo capaz de considerar informações de diversas fontes no intuito de gerar inferências que possam auxiliar na detecção de situações de interesse para aprimoramento da segurança do ambiente ubíquo monitorado. Este trabalho constitui colaboração ao *middleware* EXEHDA (LOPES et al., 2014) e derivado trabalho voltado à SI denominado EXEHDA-USM (ALMEIDA, 2016), integrando e ampliando as funcionalidades de identificação de situações de interesse no âmbito de SI, focada em ampliar as possibilidades da camada de compreensão de CS. A partir de uma perspectiva da base teórica de CS, a figura 16 situa o modelo EXEHDA-SO compondo a camada de Compreensão.

Destaca-se que na fase de percepção, os dados podem ser oriundos de diversas soluções de segurança e ativos de rede, elementos que configuram um ponto inicial do processo de sensoriamento dos eventos. Componentes básicos de rede como *Switchs, Access Points, IoT Devices,* soluções de segurança, ou seja, todo o ativo que disponha de comunicação com a rede e mínimo poder de processamento, pode ser uma fonte de contexto em potencial.

Como são previstos eventos dos mais diversos sensores com distintos formatos de dados, a Coleta de Eventos (executada tipicamente pelos componetes Colector's)

encaminha os registros brutos para a fase de Pré-processamento que caracteriza o início da fase de Compreensão. No pré-processamento ocorre a normalização, contextualização, categorização e priorização dos eventos.

Já a etapa de Processamento na fase de Compreensão é caracterizada pela atuação das técnicas de Processamento de Eventos Complexos (do inglês, Complex Event Process), e a partir da contribuição do modelo EXEHDA-SO que é apresentado neste Capítulo, também conta com uma abordagem ontológica de SI.

A camada de projeção tem a intenção de prevenir novas ocorrências de cenários indesejados que foram identificados durante a fase de compreensão. Esta conta com atuadores ativos, como *scripts* personalizados que podem gerenciar a execução distribuida promovendo modificações no ambiente, e atuadores passivos quando as contramedidas requerem uma análise prévia para execução, neste caso enviando alertas via e-mail, SMS (Short Message Service) ou abrindo um chamado em um sistema de suporte.

O Repositório Híbrido de Contexto (RHIC) proposto no trabalho (MACHADO et al., 2017) foi incorporado a este projeto, por contemplar um banco de dados não relacional e um banco voltado ao armazenamento de triplas para as instâncias do modelo ontológico. A partir deste repositório é possível consultar o histórico de ações maliciosas executadas no ambiente, já com as buscas de cenários de interesse efetuadas.

Quanto à apresentação do modelo ontológico, foi feita uma segmentação da ontologia em 3 fragmentos. O primeiro deles, denominado **Core**, intrumentaliza a busca por CS provendo alguns conceitos básicos que abrangem o domínio de SI de forma geral. Já o **Scope analyzer** executa raciocínio ontológico típico ao componente *SmartLogger*, auxiliando na identificação de cenários em célula única. Complementando a abordagem define-se o segmento **Intercell analyzer** que dispõe de uma visão geral da arquitetura multinível e valendo-se disso efetua ações preventivas visando proteção do ambiente ubíquo considerando situações identificadas nas células de distintos níveis hierárquicos, podendo atuar tanto no *SmartLogger* quanto no *Manager*.

As próximas seções abordam estes fragmentos ontológicos, relacionando os conceitos definidos em cada um. Este modelo pode ser acessado em https://gitlab.com/diorgenesyuri/EXEHDA-SO, já com algumas instâncias de exemplo, bem como regras que são tema do Capítulo 6, que trata dos casos de uso.

5.1 EXEHDA-SO: Core

O fragmento ontológico denominado **Core** visa representar conceitos pertinentes a aspectos gerais de SI e do próprio *middleware* EXEHDA, no sentido de transversalizar uma base de conhecimento em apoio as funcionalidades voltadas a CS tratadas ao longo do modelo como um todo. Este fragmento, bem como os demais,

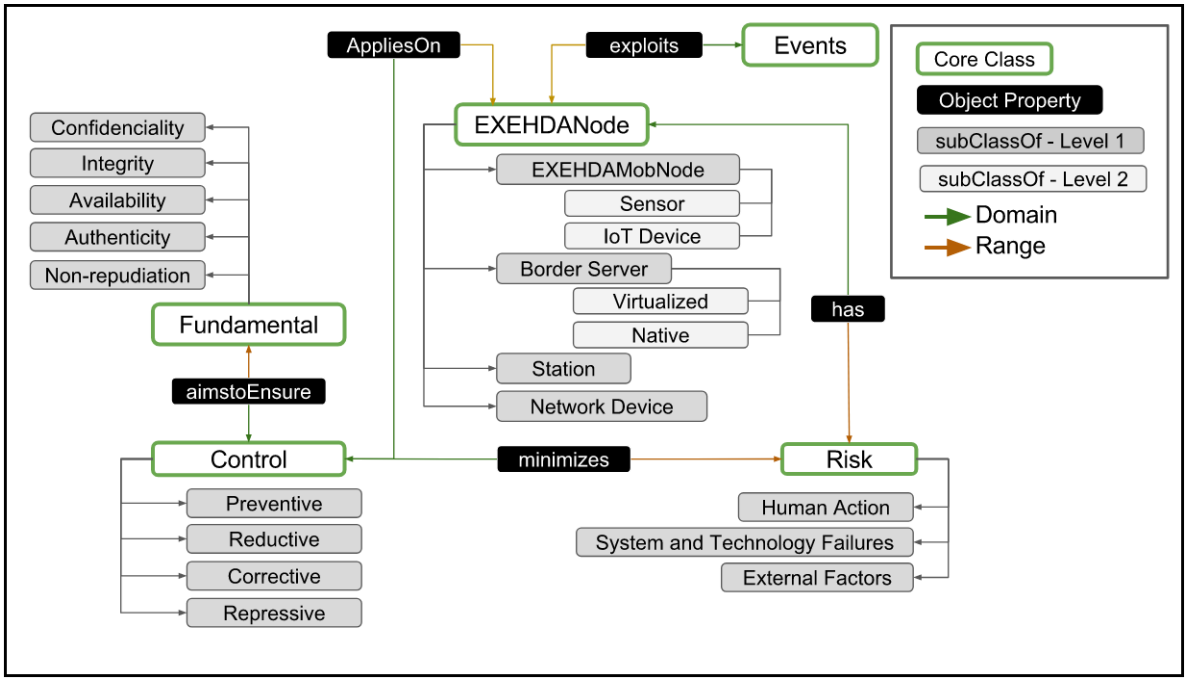

Figura 17: Principais classes do fragmento Core

consiste em uma ontologia tipificada como Leve. Não busca-se representar a completude de conceitos estabelecidos em bibliotecas do domínio de SI, refletindo sim apenas a abstração necessária, em mundo fechado, para as demandas deste projeto.

A estrutura hierárquica de vocabulário expressa neste fragmento é detalhada com o auxílio da figura 17, onde há uma demonstração dos conceitos contemplados pelas classes e seus respectivos relacionamentos por intermédio das propriedades de objeto definindo, por conseguinte, domínio e *range* das relações. Em certa medida o vocabulário escolhido mostra-se consonante às definições já presentes e documentadas ao longo do desenvolvimento do *middleware* EXEHDA, conforme pode ser verificado na mesma ilustração. Ter esta aproximação de termos como um dos focos do modelo auxilia em questões léxicas, principalmente na recorrente sinonímia de termos do domínio abordado.

5.1.1 Estrutura do Fragmento Core

As subclasses de *Fundamental* remetem aos 5 aspectos teóricos centrais de SI destacados pela ISO 27002[1]. Ações maliciosas em redes de computadores de forma geral visam comprometer pelo menos um destes aspectos e conforme indica a propriedade *aimstoEnsure*, os Controles tem por objetivo justamente garantir estes aspectos ao ambiente. São eles:

- *Confidenciality*: significando que apenas as pessoas autorizadas podem ter acesso as funcionalidades do ambiente.

[1] ISO 27002: boas práticas para a gestão de segurança da informação

- *Integrity*: referindo-se a garantia de que a informação não foi alterada de forma indevida ou não autorizada.
- *Availability*: caracterizando que a informação deve estar acessível sempre que necessário.
- *Authenticity*: propriedade complementar de SI indicando que a informação foi produzida, expedida, modificada ou destruída por uma determinada pessoa física ou por um determinado sistema, órgão ou entidade.
- *Non-repudiation*: caracterizando a capacidade do ambiente em não permitir a negação, por parte de um usuário, de uma ação.

De forma geral, os cenários de interesse são caracterizados no momento em que confirma-se ou existe a suspeita de que estão ocorrendo eventos voltados a perda de algum destes aspectos fundamentais de SI. Estes ativos que compõem o ambiente ubíquo, desde o mais básico sensor até servidores de aplicação, são passíveis de ataques e são representados no modelo pela classe *EXEHDANode*. Como subclasses desta representação tem-se (i) *EXEHDAmobNode* englobando dispositivos móveis, equipamentos específicos de IoT ou sensores diversos, (ii) *Station* para as estações de trabalho não móveis, (iii) *Border Server* para servidores distinguindo instalações nativas e virtuais. Por intermédio das propriedades de dados das instâncias da classe *EXEHDANode* é possível estabeler um perfil de execução que dita as regras pelas quais aquele ativo é submetido em termos de conectividade com a rede interna e externa, sistema operacional, hardware disponível, entre outros.

A classe *Control* representa toda a contra partida que pode ser adotada em prevenção ou correção a um determinado cenário de ataque identificado. A classe *Controls* descreve requisições de mudança no ambiente como sendo (i) *Preventive*, ao estabelecer controle voltado a alguma atuação que possivelmente pode ocorrer no ambiente observando comportamentos passados; (ii) *Reductive*, quando mesmo sem garantir a não ocorrência de determinada ação há uma diminuição das situações, minimizando as falhas decorrentes; (iii) *Repressive*, que são medidas que neutralizam as ocorrências maliciosas e (iv) *Corretive* quando são necessários ajustes estratégicos para a recuperação de falhas. Considerando que a aplicação destes controles pode impactar o ambiente de alguma maneira, são estipuladas ações ativas e passivas. Por ação ativa entende-se toda a ação voltada a proteção ao ambiente executada de forma automática, quando a criticidade do cenário impõe isto. O bloqueio de uma porta de comunicação por intermédio de um script personalizado que interage com o *Firewall* exemplifica um controle ativo. Já os controles passivos são aqueles que apenas instrumentalizam a análise de um gestor de redes. O envio de um e-mail é considerado uma

ação passiva, pois não interage diretamente com os perfis de execução do ambiente de forma automática.

Para a classe *Risk* são avaliados tanto o impacto que a exploração de determinado ativo pode representar quanto a probabilidade com que esta exploração ocorra. Cenários percebidos tem seus riscos determinados por dados contextuais que podem ser incluídos inclusive na fase de pré-processamento, como por exemplo (i) severidade da situação (ii) prioridade do evento e (iii) criticidade do ativo, sendo tratados como propriedades de dados. Como subclasses de *Risk* foram definidas:

- *Human Actions*: quando o risco decorre de uma ação humana deliberada ou não intencional.
- *System and Technology Failures*: possíveis falhas de dimensionamento ou desempenho de hardware, incompatibilidade ou más práticas de desenvolvimento de software.
- *External Factors*: possíveis incidentes naturais que possam impactar os recursos tecnológicos do ambiente lotados principalmente no datacenter.

Todo o evento percebido que possa impactar algum dos fundamentos básicos de SI estabelecidos no modelo, é representado pela classe denominada *Events*. Esta é uma das classes principais do modelo pois a partir dos eventos, geralmente estruturados em logs de serviços ou soluções de segurança, encaminha posterior identicação de situações de interesse. Cada instância desta classe trás consigo uma série de predicados, materializando dados contextuais, que definem o evento segundo alguns aspectos técnicos relevantes, como: endereçamentos IP's e *HostName* de origem e destino do evento, o serviço alvo, por exemplo. Por intermédio dos predicados das instâncias desta classe é possível verificar o método utilizado pelos atacantes, sua estratégia e intenção. Se o evento refere-se a um escaneamento de portas (atividade tipicamente preliminar à ataques), tentativa de entrega de *malware's*, tentativas de acesso ilegítmo, entre outros. Conforme demonstrado pela propriedade *"exploits"*, os eventos são ações maliciosas destinadas a *EXEHDANode's* do ambiente, sendo um vocábulo estipulado para ataques ou ações tipicamente preliminares à ataques.

5.2 EXEHDA-SO: Scope analyzer

A concepção do fragmento ontológico **Scope analyzer** define as representações necessárias às observações das situações de interesse do ambiente Ubíquo no que diz respeito às primeiras fases de processamento ontológico. O foco central deste fragmento consiste na busca de reconhecimento de contextos específicos de segurança

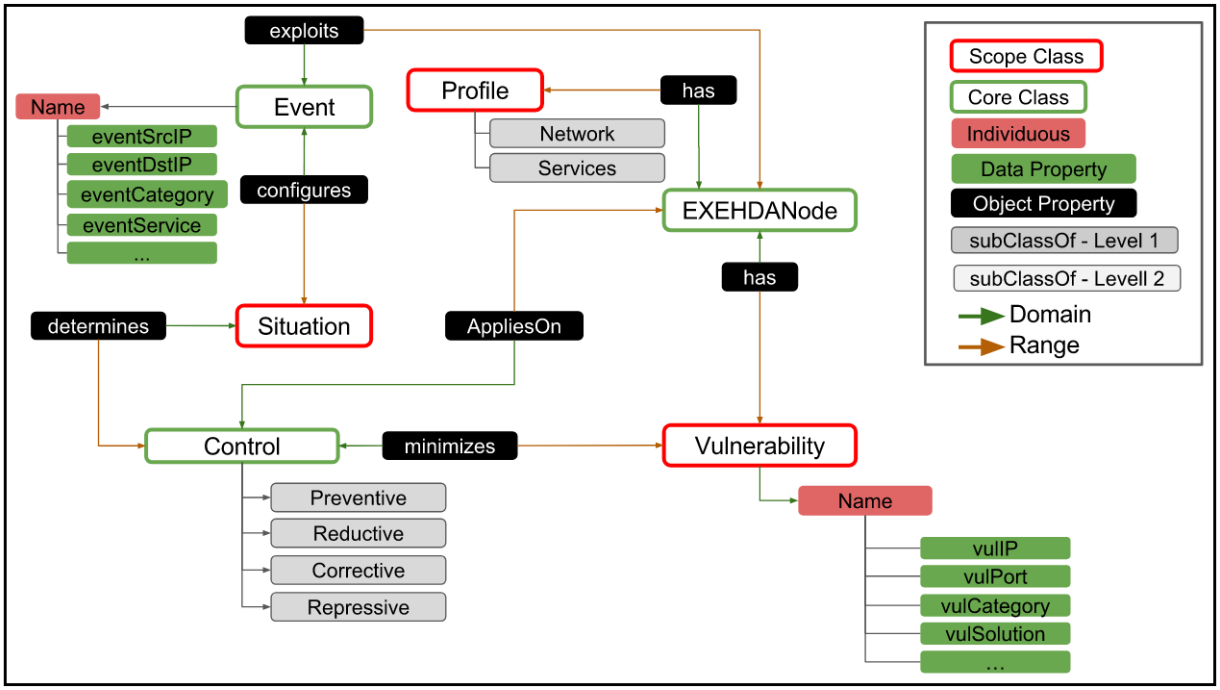

Figura 18: Principais classes do fragmento Scope analyzer

na célula de execução atribuída a ele. A figura 18 demonstra as classes e relações que este fragmento inclui ao modelo.

O processamento dos cenários já nas células aproxima a correção ou remediação dos nodos envolvidos no que tange a conectividade, isto é, a correção é aplicada sem percorrer outros níveis da arquitetura. Esta alternativa também auxilia a questão de escalabilidade no tratamento dos eventos ao distribuir o processamento entre setores distintos do ambiente.

5.2.1 Estrutura do Fragmento Scope Analyzer

As relações entre as demais classes auxiliam a definição de vulnerabilidades, representada pela classe *Vulnerability* do modelo. Neste sentido observa-se ainda na figura 18 que um controle tem por intenção a minimização de uma vulnerabilidade, podendo inclusive eliminá-la protegendo assim os *EXEHDANode's* alvo que eventualmente possuam a vulnerabilidade em questão.

Relaciona-se com *EXEHDANode* a classe denominada *Profile*. Esta representa as configurações vigentes empregadas aos recursos computacionais pondendo caracterizar regras de conectividade na subclasse *network* e quais serviços estão ativados em *Services*. Mais especificamente, a subclasse *Network* pode ter padrões permissivos ou restritivos, observada a relevância e criticidade do equipamento em questão.

A classe *Events*, descrita no fragmento **Core**, possui considerável relevância neste fragmento, uma vez que são aceitos dados oriundos de fontes distribuídas e heterogenêas. Com isso, ao identificar padrões de interesse com base nas propriedades de

dados são então configuradas as situações, representadas pela classe *Situation*, as quais por sua vez determinam o uso de um determinado controle. A classe *Situation* pode ser compreendida como um repositório de cenários de interesse.

5.3 EXEHDA-SO: InterCell Analyzer

Observando a larga conectividade e distribuição dos ambientes ubíquos pode-se considerar que uma análise de eventos individualizada das células de execução não é capaz de gerar, por si só, uma visão global do ambiente quanto a situações de SI. Ou seja, é necessário avaliar cenários de múltiplos níveis da arquitetura valendo-se assim do cruzamento de informações entre células distintas. Desta forma a intenção do fragmento **InterCell Analyzer** é prover uma visão unificada do ambiente, recebendo ocorrências situacionais de múltiplas células e inferindo controles, considerando que a ação pode ser efetuada em todo o ambiente.

Como referido anteriormente, este fragmento pode integrar tanto o componente Manager quanto o *SmartLogger* do EXEHDA-USM dependendo da abrangência dos mesmos. O **Intercell** recebe os dados tratados no fragmento **Scope Analyzer** adicionando alguns conceitos necessários a ações entre células. A figura 19 é apresentada para mostrar as principais classes e suas relações da **Intercell Analyzer**, e na seção subsequente, 5.3.1, amplia-se a discussão acerca desta estrutura.

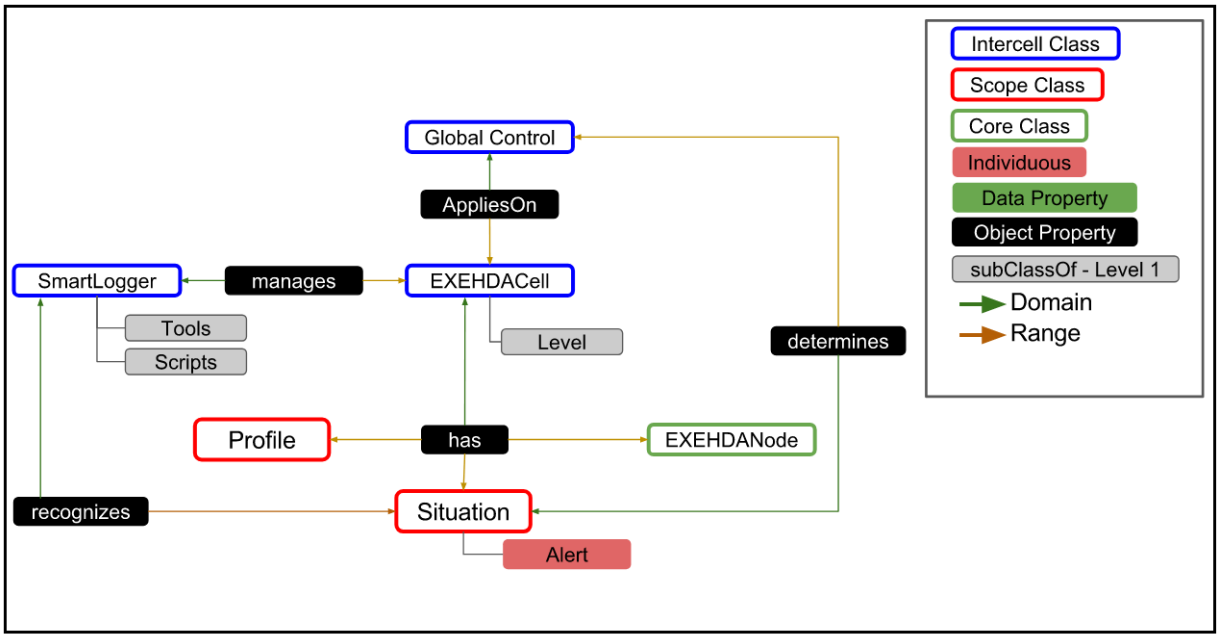

Figura 19: Principais classes do fragmento InterCell analyzer

5.3.1 Estrutura do Fragmento InterCell Analyzer

A classe *EXEHDACell* é responsável por conhecimento referente às células que compõem o ambiente ubíquo, em seus diversos níveis. Esta classe é instanciada por informações sobre os ativos presentes naquele parque computacional por intermédio da sua relação com a classe *EXEHDANode*. É então obtido um panorama geral do funcionamento das células por intermédio dos equipamentos que regem as conexões. Já na subclasse *level*, específica desta classe, é delimitada a abrangência daquela célula no que se refere a questões organizacionais e até mesmo disposição geográfica.

O componente responsável pela primeira fase de processamento, o *SmartLogger* é representado aqui por classe que leva a mesma nomenclatura. Assim a **InterCell** reconhece a gestão deste componente considerando subclasses que trazem dados sobre as técnicas de coleta utilizadas, quais são os sensores disponíveis, quais são os *scripts* personalizados existentes, entre outros. Esta classe também indica as *EXEHDACell's* nas quais o *SmartLogger* efetua o gerenciamento, podendo ser em um ou mais níveis da arquitetura.

Neste fragmento novamente a classe *Situation* está presente, contudo desta vez incorporando informações contextuais sobre a localização onde a situação ocorreu. Observar as situações instanciadas por fragmentos ontológicos dispostos em setores periféricos da arquitetura no **InterCell Analyzer** serve como um gatilho preventivo à situações identificadas em outras células do ambiente. Desta forma, identifica-se a propriedade de objetos *determines*, ligando estas situações aos *Global Controls*. Diferente da classe *Control* do fragmento de **Scope**, esta classe provê contramedidas observando todas as células. Com esta ação transversalizada no ambiente, protegem-se serviços e funcionalidades passíveis dos mesmos cenários, mesmo que as ocorrências ainda não tenham explorado àquele ambiente.

5.4 Considerações Finais

Este Capítulo pontuou a localização deste trabalho na fase de Compreensão de CS. Logo, foi apresentado o modelo EXEHDA-SO em suas particularidades de Classes e relacionamentos por intermédio de seus 3 fragmentos. Primeiramente discutiu-se os conceitos previstos no fragmento **Core** com função de trazer conceitos básicos de SI às funcionalidades dos demais fragmentos. Logo passou-se a demonstrar as classes voltadas ao âmbito celular da arquitetura no fragmento denominado **Scope analyzer**. E por fim tratou-se da hierarquia do fragmento denominado **InterCells analyzer**, que tem por intuito uma visão holística do ambiente ubíquo. Também buscou-se demonstrar a forma na qual estes fragmentos se complementam na busca de uma alternativa ontológica de processamento para eventos de SI.

Com a finalidade de validar a EXEHDA-SO foram desenvolvidos dois exemplos de

uso, apresentados no Capítulo 6, nos quais o processamento ontológico avaliou os eventos instanciados e determinou a adoção de controles para a fase de Projeção.

6 CENÁRIOS DE USO

A Universidade Federal de Pelotas (UFPel) apresenta em seu conjunto de recursos tecnológicos uma série de características comuns aos conceitos de UbiComp. Dentre estas características destaca-se a elevada distribuição na qual os recursos tecnológicos estão dispostos, isto é, configurados fisicamente. Além dos dois principais *datacenter's*, que estão localizados nos principais campus, existem outros diversos prédios que compõem o ambiente, nos quais são mantidos servidores de borda provendo filtro de pacotes, serviços de *wireless*, entre outras funcionalidades em atenção as particularidades de cada local. Também merece destaque a questão da dinamicidade de demandas tecnológicas destes locais. A conectividade é constantemente adaptada às mudanças estratégicas da instituição em termos de infraestrutura, com ampliações, adições e remoções de localidades. A arquitetura celular do EXEHDA-USM prevê esta rotina de trabalho, onde recursos são adicionados e removidos constantemente do ambiente, garantindo a modularidade necessária à organização e a autonomia de seus trechos de rede.

Os dois *datacenter's* que detêm protagonismo no parque computacional da UFPel são o do Campus Anglo (CA) e do Campus Capão do Leão (CCL). Neles são mantidos os principais serviços de TI (Tecnologia da Informação) da UFPel como o Sistema Acadêmico, Emails, Hospedagem de Sites e Sistemas de Gestão de Conteúdos (do inglês, *Content Management Systems* - CMS's), Gerenciamento de redes *Wireless*, Sistemas de Atendimentos, Ambientes Virtuais de Aprendizagem (AVA's), entre outros. Além destes serviços ainda existem diversos outros sistemas relativos a projetos específicos das unidades acadêmicas, sendo considerados Serviços de Terceiros. Também complementam o ambiente, alguns sistemas legados, para os quais não há mais atualizações ou suporte de nenhuma equipe de desenvolvimento de software, mas ainda precisam ser mantidos *online*.

Dentre os desafios oriundos deste cenário pode-se observar que dispor de recursos importantes em diversas células de execução amplia a demanda referente à análise de múltiplas ações maliciosas. Isto ocorre principalmente pela descentralização da gestão dos recursos e também pela alta frequência de eventos,

derivados tanto da rede interna quanto externa.

O caso de aplicação apresentado neste Capítulo leva em consideração estes fatores para realizar suas simulações, análises dos eventos de ataques e contramedidas. Explora-se o raciocínio ontológico para identificação de situações em níveis distintos da arquitetura, efetuando processamento distribuído, provendo prevenção para todas as células que compõem o ambiente ubíquo a partir de cenários identificados em uma única célula.

Pontua-se que, de forma análoga ao ambiente da UFPel, foram instalados os seguintes servidores, conforme a figura 20 demonstra:

- Servidores Web: nestes servidores são hospedados sites para os quais, de forma geral, não há mais manutenções ou atualizações. São projetos antigos que ainda detém grande importância institucional, contudo por questões técnicas (versões de serviços, por exemplo) e de gestão, não têm mais investimentos.
  - *webserver1* (WS_1): sistemas gerenciadores de conteúdo, do inglês *Content Management System*. O endereço IP deste ativo é 192.168.0.1.
  - *webserver2* (WS_2): hospedagem de sites com painel de controle *open source* para gerenciamento dos ambientes web. O endereço IP deste ativo é 192.168.0.2.
  - *webserver3* (WS_3): Sites descontinuados que ainda precisam estar acessíveis para consulta, entre outros sistemas legados. 192,168.0.3.
- *Vulnerability Analysis* (VA): servidores que foram distribuídos no ambientes simulados do CCL e do CA, os quais possuem um conjunto de soluções sem custo e de código aberto para análise de vulnerabilidades incluindo OpenVAS[1] (Open Vulnerability Assessment System), Nikto[2], wapiti[3], sqlmap[4], entre outros, e *scripts* personalizados para automatização de determinadas tarefas.
- *Network Intrusion Detection System* (NIDS): o suricata-ccl e o suricata-ca são responsáveis por realizar a análise do tráfego de rede da DMZ (*demilitarized zone*) nos dois campus, sendo baseados no IDS suricata[5]. Nesta avaliação será explorado especialmente o suricata-ccl, estando o suricata-ca disposto na figura 20 especificamente para fins de explicitação da infraestrutura.
- *Host Intrusion Detection System* (HIDS): Sistema de Detecção de Intrusão Baseado em *Host*, configurado em cada *webserver*, monitora o comportamento ou os

[1] http://www.openvas.org/
[2] https://cirt.net/Nikto2
[3] http://wapiti.sourceforge.net/
[4] http://sqlmap.org/
[5] http://suricata-ids.org/

estados dinâmicos de uma máquina verificando eventos registrados em log ou os dados de auditoria e, portanto, detecta e avalia as mudanças no sistema de arquivos, controle de acesso de usuários, o comportamento dos processos do sistema e o uso de recursos, entre outros. Um HIDS também realiza inspeções simples de pacotes que chegam a partir da rede em que a máquina está conectada.

- *Firewall*: foi prevista a existência de um Firewall para a DMZ de cada datacenter. Este firewall realiza a filtragem de pacotes seguindo uma política restritiva de acesso. O mesmo considera apenas o cabeçalho das camadas de rede e de transporte, camada 3 e 4 do modelo OSI (*Open System Interconnection*) respectivamente. Ou seja, as regras são formadas indicando os endereços de rede (de origem ou destino) e as portas TCP/IP envolvidas na conexão.

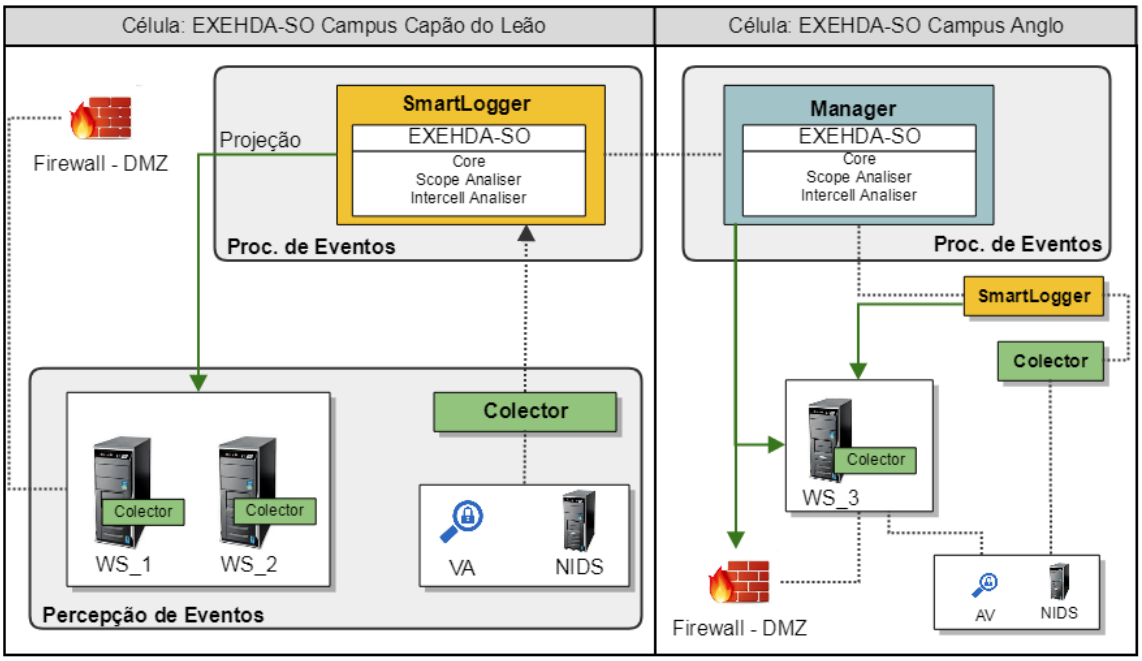

Figura 20: Componentes EXEHDA-SO: disposição de ativos em duas células

Ao fazer alusão ao parque computacional da UFPel, com disposição de serviços conforme a figura 20, o estudo de caso reflete a arquitetura celular do EXEHDA-USM (ALMEIDA, 2016). Além dos ativos previamente descritos, no ambiente de avaliação foram alocados um Colector focado na percepção e pré-processamento de eventos instalado diretamente nos servidores, junto à dois Smartlogger's, contendo as funcionalidades ontológicas iniciais, e um Manager. Com isso, é possível estabelecer a localização da abordagem EXEHDA-SO disposta nas duas diferentes células. Destaca-se assim que a alternativa de processamento ontológico soma-se as regras de processamento de eventos complexos (CEP) já disponíveis nos componentes de

sua arquitetura, ampliando a capacidade de visualização dos cenários e provendo inferências.

As simulações apresentadas a seguir ocorrem valendo-se inicialmente das funcionalidades do Colector, que recebe os eventos brutos de soluções de segurança que desempenham o papel de sensores, normalizando-os, priorizando-os e contextualizando-os. Em um segundo momento estes eventos são encaminhados ao SmartLogger, onde o módulo ontológico começa a processar estes eventos na procura de situações de interesse. Neste ponto, com os dados instanciados no SmartLogger, são possíveis diversos testes para identificação de padrões de ataques, padrões estes que indicam muitas vezes cuidados ou contramedidas interessantes à proteção do ambiente. A ontologia de segurança presente no SmartLogger efetua estes processos de inferência que serão úteis tanto para a célula na qual o SmartLogger está atuando, quanto para ações preventivas em outras células que também podem ser alvo do mesmo processo de ataques. Este processo completo entre os componentes do EXEHDA-USM pode ser visto na figura 21 onde a atuação ontológica no SmartLogger e em um segundo momento no Manager.

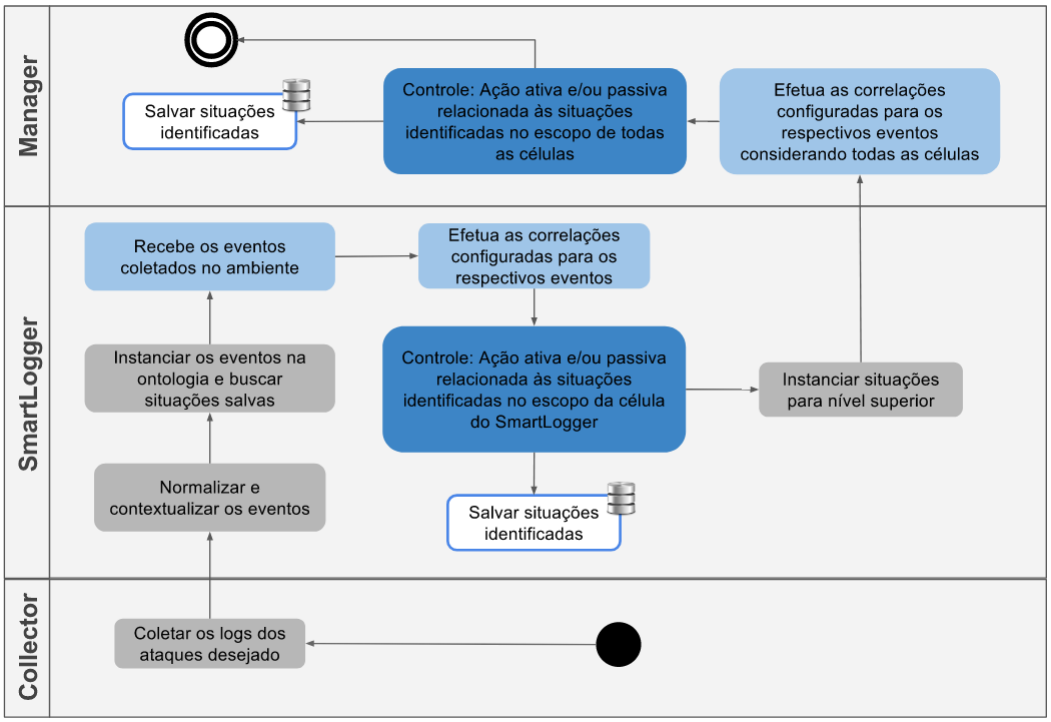

Figura 21: Fluxo entre os componentes arquiteturais e atuação ontológica

A seguir serão relacionadas algumas situações de interesse na qual as regras ontológicas foram utilizadas na busca por padrões de ataques, demonstrando as fases de CS contempladas.

6.1 Análise de Endereçamento dos Eventos

Proteger o ambiente a partir da análise de origem e destino dos eventos detectados é um dos eixos centrais do estudo de caso. Verificar se os eventos de ataque são oriundos de apenas um ou mais *hosts* trás indícios da abrangência necessária das contramedidas. Isto é, se a origem dos eventos coletados, verificada na ontologia pela propriedade *eventSrcIP*, for recorrente para um ou mais destinos é interessante que o analista considere especificamente o bloqueio daquele agente seguindo indicação de contramedida documentada na classe *Controls* da ontologia. Por outro lado, se os eventos refletem a ação de muitas origens (novamente com base no *eventSrcIP*) em um só destino (considerando o campo *eventDstIP*), a contramedida pode considerar o sugestionamento ao analista de segurança para avaliar a possibilidade de diminuir as possibilidades de ataques no servidor de borda em questão (desativando serviços, limitando o acesso aos mesmos apenas da rede interna, entre outras alternativas que objetivem a mitigação do risco).

Para demonstrar esta identificação, primeiramente o Colector recebe o evento bruto conforme simulação visualizada na figura 22. Este log é oriundo do HIDS (*Host-based Intrusion Detection System*) OSSEC *(Open Source Security)* que está configurado em modo *Standalone* no servidor WS_2. Observa-se que um evento similar a este é também registrado no WS_1 diferindo apenas o IP e *hostname* do servidor de destino e o momento da ocorrência.

```
AV - Alert - "15223568482" --> RID:"11306"; RG:"syslog,pure-ftpd,
authentication_failures,";RC:"FTP brute force (multiple failed logins).";
USER "NONE"; SRCIP:"192.168.0.10"; HOSTNAME:"(webserver2) 192.168.0.2->
/var/log/messages"; LOCATION:"(webserver2) 192.168.0.2->/var/log/messages";
EVENT: "[INIT]Set 12 15:31:40 webserver2 pure-ftpd: (?@192.168.0.10)
[WARNING] Authentication failed for user [webmaster][END]"
```

Figura 22: Log bruto identificando: evento de força bruta para PureFTP

Este evento reflete um ataque de Força Bruta (conhecidos em inglês como *Brute Force Attacks*) que consiste em uma ação maliciosa na qual é realizada a tentativa de descoberta de credenciais de um determinado sistema ou serviço, geralmente por intermédio de um dicionário *(wordlists)* ou por combinação de caracteres. Para execução do ataque em questão foi utilizada a ferramenta THC-Hydra[6] disponível nativamente na distribuição linux Kali [7].

Os *logs* que caracterizam os eventos de Força Bruta exemplificados na figura 22

[6] https://www.thc.org/thc-hydra/
[7] https://www.kali.org/

são então tratados pelo componente *Colector* dos servidores. Este estágio de pré-processamento, vale-se inicialmente do Filebeat que lê *logs* oriundos de diversos sensores do ambiente, inclusive do HIDS OSSEC. O Filebeat encaminha então os *logs* recebidos ao Logstash, controlando o fluxo deste encaminhamento para que não ocorra sobrecarga, provendo métodos de retomada do encaminhamento em caso de falha do Logstash e criptografando esta comunicação. No Logstash estes dados são normalizados utilizando a expressão regular que pode ser vista na figura 23.

```
# OSSEC - alerts.log
OSSECALERT AV - Alert - "%{NUMBER:timestamp}" --> RID: "%{POSINT:rule_id:int}";
RL: "%{POSINT:rule_level:int}"; RG: "%{DATA:rule_group}"; RC: "%
{DATA:rule_comment}"; USER: "(%{DATA:user}|\(no user\))"; SRCIP: "(%{IP:src_ip}|%
{HOSTNAME:src_hostname}|None)"; HOSTNAME: "(%{HOSTNAME:hostname}|\(%
{HOSTNAME:hostname}\) %{IP}->(%{PATH}|%{NOTSPACE}))"; LOCATION: "(%
{PATH:location}|%{NOTSPACE:location}|\(%{HOSTNAME}\) %{IP}->(%{PATH:location}
|%{NOTSPACE:location}))"; EVENT: "\[INIT\]%{DATA:event}\[END\]";
```

Figura 23: Expressão regular Logstash para registros OSSEC

O resultado da normalização do Logstash sobre os dados dos eventos são disponibilizados em JSON, conforme demonstrados na figura 24. Ainda nesta figura, observa-se a adição dos campos *category*, *sub_category* e *priority*, os quais são resultantes da contextualização empregada pelo Logstash. A partir deste estágio de pré-processamento já se pode visualizar de maneira simplificada, quando comparado ao evento bruto, algumas das informações centrais para a definição do cenário de interesse como: (i) quem executou a ação maliciosa no campo SRCIP; (ii) o alvo escolhido pelo atacante no campo DSTIP; (iii) quando ocorreram as ações no campo ”date”; (iv) o serviço relacionado e o tipo do ataque no campo “rule_comment”.

Após executadas estas tarefas, o componente Colector encaminha os eventos para o SmartLogger da célula que representa o CCL, onde a ontologia é instanciada na classe *Events* para ataques, conforme as triplas demonstradas na tabela 3. Observa-se que nem todos os campos foram encaminhados para a ontologia, sendo utilizados apenas os dados relevantes para o processamento ontológico deste caso. Para efetuar o encaminhamento dos eventos na ontologia foi utilizada a API Java Jena.

Neste ponto, com o evento apresentado na tabela 3 em conjunto com o evento similar produzido no WS_1 já instanciados na classe estipulada *Events*, tem-se o conhecimento de diversos fatores fundamentais de uma ação maliciosa como (i) quais foram os endereçamentos de origem e destino do evento; (ii) o serviço alvo; (iii) o usuário que está sendo utilizado pelas tentativas de acesso e (iv) a prioridade do evento.

A estratégia para esta identificação de cenário primeiramente separa os eventos por intermédio do predicado destino denominado *eventDstIP* que informa o endereço IP alvo. Assim, na simulação realizada, todos os eventos destinados para o WS1 são

{_id: ObjectId("5579fbd26e58bc7ac47bb87"),
date: "2017-09-12 15:31:40.000000";
rule_id:"11306",
rule_level:"10",
rule_group:['syslog','pure-ftpd','authentication_failures'],
rule_comment:"FTP brute force (multiple failed logins)",
srcip:"192.168.0.10",
username:"webmaster",
hostname"webserver2",
location:"/var/log/messages",
dstip:"192.168.0.2",
event:"Set 12 15:31:40" webserver2 pure-ftpd:(?@192.168.0.10)
[WARNING]
Authentication failed for user [webmaster]".
category:"authentication",
sub_category: "failed",
priority: "10" }

Figura 24: Pré-processamento: normalização de eventos no Logstash

Tabela 3: Triplas da classe *Events*

Sujeito	Predicado	Objeto
Events	eventSrcIP	192.168.0.10
Events	eventDstIP	192.168.0.2
Events	eventCat	authentication
Events	eventService	FTPD
Events	eventDate	2017-09-12
Events	eventSUser	webmaster
Events	eventDstHN	webserver1
Events	eventTime	15:31:40
Events	eventName	Bruteforce

encaminhados para a classe que o representa na hierarquia de ativos *(EXEHDANode)*. Esta regra para o WS1 pode ser visualizada na figura 25. A regra repete-se para cada Web Server previsto.

```
exehdaso:Events(?x) ^ exehdaso:eventDstIP(?x, ?a) ^ swrlb:equal(?a,
"192.168.0.1") ^ exehdaso:eventName(?x, ?b) ^ swrlb:equal(?b,
"bruteforce") -> exehdaso:BF_WS_1(?x)
```

Figura 25: Regra para encaminhar os eventos para a classe que representa o ativo alvo

A tarefa de perceber que muitos eventos apresentam uma mesma origem para mais de um destino no ambiente é executada pela regra da figura 26. Nesta regra SWRL executada pelo SmartLogger do CCL, a origem de eventos identificados nos dois servidores web são comparados e caso apresentem a situação de interesse, estes eventos são então instanciados na classe *Situation*, a qual por sua vez determina os controles a serem identificados por meio da classe *Control*.

```
exehdaso:BF_WS_1(?x) ^ exehdaso:BF_WS_2(?y) ^ exehdaso:eventSrcIP
(?x, ?a) ^ exehdaso:eventSrcIP(?y, ?b) ^ swrlb:equal(?a, ?b) ^
exehdaso:eventDstIP(?x, ?c) ^ exehdaso:eventDstIP(?y, ?d) ^
swrlb:notEqual(?c, ?d) -> OSND(?x) ^ OSND(?y)
```

Figura 26: Regra para identificação de origens únicas em muitos destinos

A partir deste momento, a classe *Situation* é instanciada com os eventos cujas propriedades de dados informam quais endereços IP's estão envolvidos no cenário e a ação, que implica na necessidade de adaptação do ambiente ubíquo promovendo o bloqueio do IP do atacante. Neste momento, é inferido que a recorrência deste atacante no ambiente indica a possibilidade do mesmo realizar novas tentativas de acesso indevido nesta célula. Considerando isto, o *Control* da EXEHDA-SO Core sugestiona que o bloqueio ocorra no Firewall - DMZ, protegendo desta maneira todos os servidores de borda sob a política da DMZ. Na sequência, ocorre a comunicação via API Jena com o módulo de projeção, o qual é encarregado de aplicar a ação ativa de bloqueio do endereçamento de origem (192.168.0.10) destas ações maliciosas no Firewall - DMZ do CCL. A figura 27 mostra as classes do modelo ontológico voltado para este cenário.

Ainda na perspectiva de análise de endereçamentos, outro cenário avaliado foi a identificação de ataques com múltiplas origens focadas em um único servidor de borda, percebendo alvos recorrentes no ambiente. Para realização deste ataque,

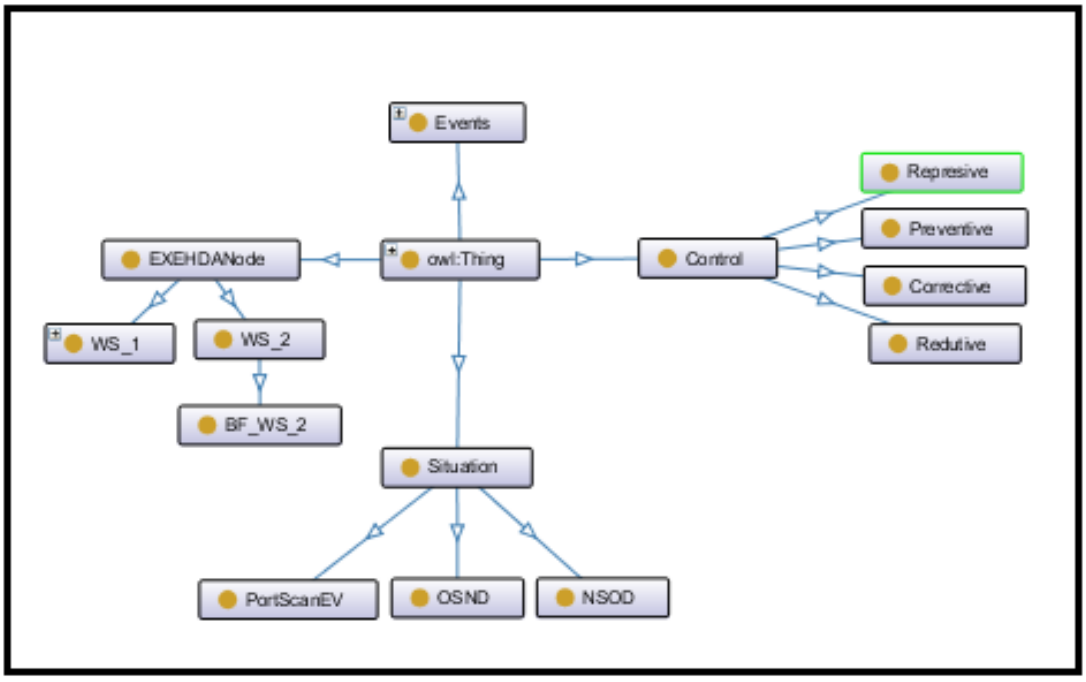

Figura 27: Classes envolvidas no processo de análise de endereçamento

foram utilizadas 2 máquinas virtuais com kali linux novamente explorando a ferramenta THC Hydra. Desta vez, como alvo do ataque foi definido o servidor webserver3 (WS_3).

Os dados dos eventos são instanciados na classe *Events*, onde posteriormente são separados por regra semelhante a anteriormente apresentada na figura 25). Na sequência, para a análise deste cenário foi estipulada a regra conforme figura 28, a qual executa no SmartLogger do CA. Como consequência da execução desta regra, ocorre a instanciação da classe *Situation*, a qual possui como controle a ação passiva de sugestionamento da desativação do usuário sob ataque (webmaster). O módulo de projeção, ao comunicar-se via API Jena com a EXEHDA-SO, realiza o envio de e-mail com os dados do controle sugerido, contribuindo para a tomada de decisões do analista. Por fim, estes eventos são salvos em um repositório para posterior análise utilizando base de dados Virtuoso.

```
exehdaso:BF_WS_3(?x) ^ exehdaso:BF_WS_3(?y) ^
exehdaso:eventSrcIP(?x, ?a) ^ exehdaso:eventSrcIP(?y, ?b) ^
swrlb:notEqual(?a, ?b) -> exehdaso:NSOD(?x)
```

Figura 28: Exemplo de regra para identificação de múltiplas origens em um destino

Uma vez concluído o processamento ontológico dos eventos no Smartlogger, este

envia os seus alertas para o Manager, componente disposto na célula alusiva ao Campus Anglo (CA). Conforme referido anteriormente, com a atuação ontológica em um nível superior da arquitetura, o Manager recebendo as situações detectadas pelos SmartLogger's de todas as células que compõem o ambiente, torna-se capaz de inferir ações de proteção preventiva para o ambiente, considerando as detecções individuais das células. Por exemplo, considerando que ambas as situações de interesse identificadas possuem como endereço IP do atacante 192.168.0.10, é possível inferir que o atacante irá continuar a sua busca por acesso indevido a outros serviços que dão suporte ao ambiente ubíquo. Neste caso, a classe de controles global da InterCell Analyzer pode indicar o processo de bloqueio do endereço em questão no firewall das diferentes células sob coordenação do Manager.

Aplicar o raciocínio sobre as classes, com o vocabulário previamente determinado, torna a busca de padrões mais genérica o que atribui ao EXEHDA-USM uma amplitude maior quanto a identificação de cenários. Como resultado desta análise busca-se uma otimização na identificação de cenários auxiliando os analistas nas tomadas de decisões dirigidas a proteção das informações trafegadas no ambiente.

6.2 Contextos Independentes

Um dos problemas anteriormente levantados nesta dissertação é a necessidade de implantação de diferentes soluções de segurança de propósitos específicos, fator que dificulta a visão integrada das mesmas. Neste segundo caso, busca-se avaliar contextos oriundos de soluções que não possuem integração nativa e cujo formato dos eventos são heterogêneos.

Para a validação desta correlação foi utilizado o OpenVAS que é uma das soluções empregadas pelo servidor VA para identificação de vulnerabilidades e o NIDS Suricata para identificação das atividades maliciosas no ambiente. Já para a simulação do ataque foi explorado o Nmap[8]. Para além da perspectiva de contextos independentes, a determinação deste cenário é interessante pois muitas vezes as equipes não realizam o tratamento das vulnerabilidades por diversas razões. Adicionalmente, o número de alertas gerados pelo Suricata quando empregando regras de detecção de varreduras de redes, pode ser consideravelmente elevado. Então, notificar um incidente associando a vulnerabilidade com uma tentativa de exploração da mesma fornece maior informação para a priorização das adaptações necessárias, novamente auxiliando a tomada de decisões.

Inicialmente é realizada uma varredura de vulnerabilidades pelo OpenVAS. O componente Colector que opera no servidor VA realiza o pré-processamento e o encaminhamento dos eventos e vulnerabilidades identificadas para o SmartLogger. Por

[8]https://nmap.org/

sua vez, é realizada a instanciação das vulnerabilidades na classe *Vulnerability* da EXEHDA-SO.

Na sequência, a ferramenta nmap, comumente empregada na primeira etapa de um teste de intrusão, é utilizada, para levantamento de informações sobre o alvo, no caso, o servidor webserver1 (WS_1). Uma das características dessa ferramenta é que ela fornece dados sobre as versões de serviços que estão em execução no dispositivo alvo. Logo, de posse dessa informação, o atacante pode valer-se das vulnerabilidades documentadas relativas às versões identificadas, especialmente se houver algum serviço desatualizado.

Como consequência desta execução, o NIDS Suricata registra os eventos que identificam esta tentativa de escaneamento de portas. Este registro de eventos emprega o formato JSON, o qual facilita a etapa de coleta por ser nativamente suportado pelo Filebeat. Além disso a etapa de pré-processamento é desonerada da normalização. Com isso, após contextualizados, os eventos são instanciados dentro da EXEHDA-SO na classe *Events* com dados que incluem origem e destino de endereçamentos, informações sobre a regra do Suricata que detectou o evento, entre outros.

Contado com estas informações de contextos independentes instanciadas nas classes *Vulnerability* e *Events* é aplicada uma regra apresentada na figura 29 que gera novas instâncias de alertas nas subclasses de *Situation*, que informam sobre a existência de uma situação de levantamento de informações de segurança em um serviço com versão exposta.

```
exehdaso:Vulnerability(?y) ^ exehdaso:Events(?x) ^ exehdaso:eventDstIP
(?x, ?a) ^ exehdaso:vulIP(?y, ?b) ^ swrlb:equal(?a, ?b) ^
exehdaso:eventDstPort(?x, ?c) ^ exehdaso:vulPort(?y, ?d) ^ swrlb:equal
(?c, ?d) -> exehdaso:PortScanEV(?y) ^ exehdaso:PortScanEV(?x)
```

Figura 29: Componentes EXEHDA-SO: disposição de ativos em duas células

Estes alertas são utilizados pela etapa de Projeção, para envio de email para o analista onde é sugerido controle já documentado no campo solução proveniente da varredura executada pelo OpenVAS. Neste caso específico é sugerida a alteração dos parâmetros de configuração do Apache *ServerTokens* para Prod e *serverSignature* para Off.

6.3 Considerações Finais

Este capítulo buscou demonstrar a potencialidade do uso de ontologias na obtenção de ciência de situação em segurança da informação. Para tal, simulou-

se ambiente semelhante ao da UFPel, por apresentar muitas das características que ainda constituem desafios da UbiComp. Inicialmente o caso de uso avaliou os eventos coletados no ambiente quanto a origem e destino, encaminhando contramedidas aos cenários identificados, no intuito de demonstrar como as regras ontológicas podem contribuir para soluções mais abrangentes. Logo, cruzaram-se os dados dos eventos com a análise de vulnerabilidades, gerando uma visão unificada de eventos de diferentes soluções que, por intermédio do raciocínio ontológico, puderam colaborar de forma integrada para a adaptação do ambiente.

Dentre os benefícios obtidos com o desenvolvimento destes cenários de uso foi possível validar:

- Suporte a heterogeneidade das fontes contextuais: os eventos de segurança podem ser oriundos de diversas soluções, que por sua vez disponibilizam dados em formatos diferentes.
- Consideração de contextos independentes: contando com os dados centralizados na ontologia é possível considerar diversos elementos que antes não seriam cruzados.
- Possibilidade de reuso: a EXEHDA-SO foi proposta tendo por base diferentes modelos ontológicos explorando a característica de reusabilidade das ontologias. A partir de então é possível que a EXEHDA-SO seja também reutilizada, adaptada e evoluída por outros projetos.

7 CONSIDERAÇÕES FINAIS

Este Capítulo relaciona as principais conclusões decorrentes do trabalho de pesquisa desenvolvido nesta dissertação, o qual resultou na EXEHDA-SO A EXEHDA-SO enquanto alternativa para Ciência de Situação (CS) no domínio de Segurança da Informação (SI). Também listam-se as publicações e oportunidades de trabalhos futuros identificadas ao longo deste processo.

7.1 Conclusões

No momento em que a informação passa a determinar desde vantagens estratégicas de mercado, direcionamentos políticos e inclusive guerras ao redor do mundo, é natural que a segurança destes dados passe a ser uma preocupação cada vez maior para todos. Este protagonismo da informação no cotidiano social é impulsionado pela materialização da UbiComp, que insere funcionalidades e serviços de forma transparente, atribuindo conectividade inclusive à dispositivos que tradicionalmente não contariam com nenhum tipo de inteligência computacional. Esta circunstância da UbiComp, que explicita o paradigma computacional denominado Internet das Coisas, propicia constantes cenários de exposição e riscos às informações.

Para que as premissas de UbiComp e IoT relacionadas sejam atendidas é requerido que os ambientes (i) garantam conectividade para um elevado número de nodos; (ii) processem e validem contextos de segurança; e (ii) sejam constantemente adaptados considerando as identificações de ações maliciosas. Voltadas para estes desafios, são recorrentemente implementadas diversas soluções focadas na estruturação de linhas de defesa de redes. Entretanto, estas soluções, por não terem finalidades iguais, muitas vezes não apresentam formatos de dados compatíveis entre sí, o que não significa que estas soluções não possam contribuir mutuamente para a construção de contexto, mas que dificulta o seu uso integrado. Sem uma constante avaliação sobre as métricas obtidas por essas soluções e sobre o tráfego que está sendo gerado, os controles implementados tendem a não alcançar o êxito esperado.

O exercício de adequação a cenários amplamente mutáveis e dinâmicos requer

mecanismos voltados a CS onde identifica-se, para além da heterogeneidade das fontes de dados contextuais, oportunidades de contribuições direcionadas à distribuição da computabilidade, à diversidade de formatos nos eventos de SI disponíveis e a multiplicidade de situações que podem ser extraídas ao combinar eventos deste perfil.

Desta forma, em resposta a estas demandas, optou-se pelo uso de ontologias, dentre outras técnicas de processamento de contexto existentes, por proverem: (i) a análise de eventos independentemente dos formatos utilizados pelas soluções de fins específicos de SI; (ii) o reuso de conhecimento, facilitando a evolução das soluções, permitindo aproximação dos vocabulários ao *middleware* para o qual este trabalho traz contribuição e auxiliando na interoperabilidade (diferentes aplicações podem valer-se dos conceitos e relacionamentos definidos nas ontologias); e (iii) a visão unificada das situações identificadas em diferentes cenários de SI.

Por intermédio de ontologias, a EXEHDA-SO integrou-se à fase de processamento de eventos de segurança da EXEHDA-USM, contribuindo para a composição de uma estratégia híbrida voltada a camada de compreensão de CS. A alternativa integra-se tanto ao componente SmartLogger quanto ao Manager, que orquestram uma arquitetura multi-nível, que é observada em diversas organizações atualmente.

A estruturação da ontologia foi desenvolvida primeiramente relacionando alguns conceitos fundamentais de SI, os quais foram estipulados no fragmento denominado Core. Este fragmento transversaliza os demais, servindo como base para todo o conhecimento que é processado na ontologia. Já o fragmento denominado Scope Analyzer buscou representar o conhecimento necessário tipicamente ao Smartlogger, encaminhando adaptações aos cenários identificados com uma maior proximidade ao trecho da arquitetura em questão. O modelo é então complementado pelo fragmento InterCell Analyzer, que visa relacionar os acontecimentos entre as diversas células que compõem o ambiente.

O estudo de caso buscou validar a estruturação ontológica apresentada no modelo, implementando mecanismos que possibilitaram a identificação de padrões de ataques configurando cenários de risco, explorando os problemas elencados na motivação desta dissertação. Para tal, fez-se uso de técnicas já consolidadas no âmbito das ontologias, como consultas em SparQL e regras SWRL. Os testes do modelo foram executados em ambiente inspirado na Universidade Federal de Pelotas (UFPel), por possuir diversas características consonantes aos desafios de UbiComp, como elevada distribuição e heterogeneidade. Foram processados eventos coletados no ambiente quanto as suas origens e destinos, encaminhando contra-medidas aos cenários identificados, no intuito de demonstrar como as regras ontológicas podem contribuir para soluções mais abrangentes. Logo, cruzaram-se os dados dos eventos com a análise de vulnerabilidades, gerando uma visão sobre a criticidade das ações maliciosas no ambiente, e novamente indicando possíveis ações preventivas.

7.2 Principais Contribuições

A estratégia apresentada nesta dissertação contribui para com a abordagem anteriormente desenvolvidas pelo grupo de pesquis, provendo vocabulário que recebe instâncias de dados contextuais de diversas fontes, processando-as de forma descentralizada, unificando formatos e provendo indicações de contra-medidas passivas ou promovendo adaptação do ambiente ubíquo por intermédio de ações ativas.

Com intuito de comparar a EXEHDA-SO com as demais abordagens descritas no Capítulo 4, apresenta-se a tabela 4. Quanto ao formalismo, as consultas e a interação foi mantido um alinhamento com os trabalhos relacionados. Entretanto, destaca-se que a EXEHDA-SO não limita-se a instâncias de uma determinada solução, como pode-se ver na coluna Eventos. Neste quesito observa-se que alguns trabalhos relacionados não informam as fontes de seus eventos enquanto outras utilizam apenas uma. A EXEHDA-SO propõe o uso de regras SWRL, as quais são mencionado em apenas um trabalho relacionado. Registra-se ainda que o uso de CEP explorado no EXEHDA é compatível com a premissa operacional da EXEHDA-SO.

Com relação às publicações, no decorrer do trabalho foram efetivadas contribuições parciais deste e de demais trabalhos do grupo relacionados ao tema dessa dissertação em diversos meios relevantes a área de Computação. Nas próximas subseções são elencadas as principais publicações obtidas.

Tabela 4: Comparação entre trabalhos relacionados e a EXEHDA-SO

	Formalismo	Consultas	Inferências	Regras	Repositório	Eventos	Interação
MARTIMIANO; MOREIRA, 2006	OWL	SparQL	-	-	MySQL	SNORT	Java Jena
AZEVEDO et al. 2009	RDF/OWL	SparQL	Pellet	-	-	-	Java Jena
BHANDARI; GUJRAL, 2014	OWL	-	Hermit	SWRL	-	-	-
AZKIA et al., 2014	RDF/OWL	SparQL	-	-	-	IHE-ATNA	-
FICCO; ROMANO, 2011	-	-	-	CEP	-	Prelude	-
EXEHDA-SO	OWL	SparQL	Pellet	CEP e SWRL	Virtuoso	Diversos	Java Jena

7.2.1 Trabalhos Publicados

- MACHADO, ROGER DA SILVA; ALMEIDA, RICARDO BORGES; **DA ROSA, DIÓRGENES YURI LEAL**; LOPES, JOÃO LADISLAU BARBARÁ; PERNAS, ANA MARILZA; YAMIN, ADENAUER CORRÊA. EXEHDA-HM: A compositional approach to explore contextual information on hybrid models. Future Generation Computer Systems, v. 73, p. 1-12, 2017.

- **ROSA, D. Y. L.** ; ALMEIDA, R. B. ; MACHADO, R. S. ; YAMIN, A. C. ; PERNAS, A. M. . UMA ABORDAGEM ONTOLÓGICA PARA CIÊNCIA DE SITUAÇÃO NO DOMÍNIO DE SEGURANÇA DA INFORMAÇÃO. In: Encontro de Pós-Graduação da Universidade Federal de Pelotas, 2016, Pelotas. XVIII ENPOS, 2016.

- MACHADO, R. S.; ALMEIDA, R. B. ; **ROSA, D. Y. L.** ; PERNAS, A. M. ; YAMIN, A. C. . UMA ABORDAGEM COMPOSICIONAL PARA CORRELACIONAR INFORMAÇÕES CONTEXTUAIS PRESENTES EM MODELOS HÍBRIDOS. In: Encontro de Pós-Graduação da Universidade Federal de Pelotas, 2016, Pelotas. XVIII ENPOS, 2016.

- MACHADO, R. S.; ALMEIDA, R. B.; **ROSA, D. Y. L.**; DONATO, L. M.; PERNAS, A. M.; YAMIN, A. C. . [SA]2: uma abordagem consciente de situação para segurança em infraestruturas computacionais. Revista Brasileira de Computação Aplicada, v. 8, p. 89-103, 2016.

- ALMEIDA, R. B.; MACHADO, R. S.; **ROSA, D. Y. L.**; DONATO, L. M.; YAMIN, A. C.; PERNAS, A. M. . Uma Arquitetura Hierárquica Multinível para Ciência de Situação em Segurança da Informação. In: Congresso da Sociedade Brasileira de Computação, 2016, Porto Alegre. SBCUP - Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2016.

- MACHADO, R. S.; ALMEIDA, R. B. ; **ROSA, D. Y. L.** ; LOPES, João L. B. ; PERNAS, A. M. ; YAMIN, A. C. . Explorando Modelos Contextuais Híbridos: uma Abordagem Composicional. In: Congresso da Sociedade Brasileira de Computação, 2016, Porto Alegre. SBCUP - Simpósio Brasileiro de Computação Ubíqua e Pervasiva, 2016.

- ALMEIDA, R. B. ; MACHADO, R. S. ; **ROSA, D. Y. L.** ; RIPPEL, H. V. ; DONATO, L. M. ; YAMIN, A. C. ; PERNAS, A. M. . NS2A: consciência de situação aplicada a segurança de redes de computadores. In: Escola Regional de Redes de Computadores, 2015, Passo Fundo. Escola Regional de Redes de Computadores, 2015. v. 13.

- ALMEIDA, R. B.; MACHADO, R. S.; **ROSA, D. Y.**; DAVET, P. T.; DONATO, L.; YAMIN, Adenauer C.; PERNAS, A. M. Segurança na Internet das Coisas: uma abordagem de SIEM customizável e baseada em Consciência de Situação. In: XLII Seminário Integrado de Software e Hardware, 2015, Recife - PE. (SEMISH 2015).

- MACHADO, R. S.; ALMEIDA, R. B. ; **ROSA, D. Y. L.** ; RIPPEL, H. V. ; YAMIN, A. C. ; PERNAS, A. M. . DLNA-ML: Uma Abordagem de Análise Dinâmica de Log e

Tráfego da Rede. In: Escola Regional de Redes de Computadores, 2015, Passo Fundo. Escola Regional de Redes de Computadores, 2015. v. 13.

- ALMEIDA, R. B. ; MACHADO, R. S. ; **ROSA, D. Y. L.** ; DAVET, P. T. ; DONATO, L. M. ; YAMIN, A. C. ; PERNAS, A. M. . Segurança na Internet das Coisas: uma abordagem de SIEM customizável e baseada em Consciência de Situação. In: Seminário Integrado de Software e Hardware, 2015, Recife-PE. Seminário Integrado de Software e Hardware, 2015. v. 42º.

- **ROSA, D. Y. L.** ; RAMBO, I. J. ; MACHADO, R. S. ; ALMEIDA, R. B. ; RIPPEL, H. V. ; Yamin, Adenauer C. ; PERNAS, A. M. . Uma abordagem híbrida para armazenamento de dados de contexto no EXEHDA. In: Escola Regional de Redes de Computadores, 2015, Passo Fundo. Escola Regional de Redes de Computadores, 2015. v. 13.

- ALMEIDA, R. B. ; MACHADO, R. S. ; **ROSA, D. Y. L.** ; DONATO, L. M. ; Yamin, Adenauer C. ; PERNAS, A. M. . Uma Proposta Distribuída para Detecção de Intrusão, Hierárquica, Multiagente e Consciente de Situação. In: Escola Regional de Alto Desempenho, 2015, Gramado/RS. Escola Regional de Alto Desempenho, 2015. v. XV.

- ALMEIDA, R. B. ; MACHADO, R. S. ; **ROSA, D. Y. L.** ; DONATO, L. M. ; YAMIN, A. C. ; PERNAS, A. M. . UMA ARQUITETURA HIERÁRQUICA MULTINÍVEL PARA CONSCIÊNCIA DE SITUAÇÃO EM SEGURANÇA DE AMBIENTES COMPUTACIONAIS. In: Encontro de Pós-Graduação UFPel, 2015, Pelotas. Encontro de Pós-Graduação UFPel, 2015. v. XVII.

- **ROSA, D. Y. L.** ; RAMBO, I. J. ; MACHADO, R. S. ; ALMEIDA, R. B. ; RIPPEL, H. V. ; YAMIN, A. C. ; PERNAS, A. M. . Uma Proposta para Gerenciamento Híbrido de Dados de Contexto no EXEHDA. In: Encontro de Pós-Graduação UFPel, 2015, Pelotas. Encontro de Pós-Graduação UFPel, 2015. v. XVII.

- MACHADO, R. S.; ALMEIDA, R. B. ; **ROSA, D. Y. L.** ; YAMIN, A. C. ; PERNAS, A. M. . Análise de Log: Uma Abordagem Baseada em Mineração Dinâmica de Dados. In: Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul, 2014, Alegrete, RS. Escola Regional de Alto Desempenho do Rio Grande do Sul, 2014. v. XIV.

- ALMEIDA, R. B. ; MACHADO, R. S. ; **ROSA, D. Y. L.** ; DONATO, L. M. ; YAMIN, A. C. ; PERNAS, A. M. . Explorando a Consciência de Situação no Gerenciamento de Aspectos de Segurança em Sistemas Distribuídos. In: Escola Regional de Alto Desempenho do Estado do Rio Grande do Sul, 2014, Alegrete, RS. Escola Regional de Alto Desempenho do Rio Grande do Sul, 2014. v. XIV.

7.3 Trabalhos Futuros

Como referido anteriormente, a área de SI mostra-se dinâmica em seus diversos aspectos, desde políticas de segurança até requisitos técnicos. Na medida em que surgem novas funcionalidades e tecnologias, também surgem novas ameaças. Desta forma, o desenvolvimento do trabalho apresentado nesta dissertação enseja a continuidade da pesquisa sob diversos aspectos. Dentre estes, destacam-se:

- ampliação dos cenários de uso por intermédio de revisão e possíveis acréscimos ao vocabulário atual.
- aumento no número de regras a serem aplicadas em busca de diferentes contextos.
- desenvolvimento de módulo voltado para adaptação e manutenção da ontologia.
- explorar a interação entre as técnicas Processamento de Eventos Complexos (CEP) e Ontologias.

REFERÊNCIAS

ALMEIDA, R. B. **EXEHDA-USM**: uma arquitetura hierárquica multinível consciente de situação aplicada a segurança da informação. 2016. Dissertação de Mestrado em Ciência da Computação — Programa de Pós-Graduação em Computação/UFPel, Pelotas - RS.

ATZORI, L.; IERA, A.; MORABITO, G. The Internet of Things: A Survey. **Comput. Netw.**, New York, NY, USA, v.54, n.15, p.2787–2805, Oct. 2010.

AZEVEDO, R. R. de; DANTAS, E. R. G.; SANTOS, R. C. dos; RODRIGUES, C.; FREITAS, F.; SIQUEIRA, M. J. An autonomic multiagent system ontology-based for management of security of information. In: INTERNATIONAL CONFERENCE FOR INTERNET TECHNOLOGY AND SECURED TRANSACTIONS, (ICITST), 2009., 2009. **Anais...** [S.l.: s.n.], 2009. p.1–9.

AZEVEDO, R. R. de; FREITAS, F.; ALMEIDA, S. C. de; ALMEIDA, M. J. S. C.; BARROS C. FILHO, E. C. de; VERAS, W. C. CoreSec: an ontology of security aplied to the business process of management. In: EATIS, 2008. **Anais...** [S.l.: s.n.], 2008.

AZKIA, H.; CUPPENS-BOULAHIA, N.; CUPPENS, F.; COATRIEUX, G. Log content extraction engine based on ontology for the purpose of a posteriori access control. **IJKL**, [S.l.], v.9, n.1/2, p.23–42, 2014.

BASS, T. Intrusion Detection Systems and Multisensor Data Fusion. **Commun. ACM**, New York, NY, USA, v.43, n.4, p.99–105, Apr. 2000.

BERNARDOS, A.; TARRIO, P.; CASAR, J. A data fusion framework for context-aware mobile services. In: MULTISENSOR FUSION AND INTEGRATION FOR INTELLIGENT SYSTEMS, 2008. MFI 2008. IEEE INTERNATIONAL CONFERENCE ON, 2008. **Anais...** [S.l.: s.n.], 2008. p.606–613.

BETTINI, C.; BRDICZKA, O.; HENRICKSEN, K.; INDULSKA, J.; NICKLAS, D.; RANGANATHAN, A.; RIBONI, D. A Survey of Context Modelling and Reasoning Techni-

ques. **Pervasive Mob. Comput.**, Amsterdam, The Netherlands, The Netherlands, v.6, n.2, p.161–180, Apr. 2010.

BHANDARI, P.; GUJRAL, M. Ontology based approach for perception of network security state. In: ENGINEERING AND COMPUTATIONAL SCIENCES (RAECS), 2014 RECENT ADVANCES IN, 2014. **Anais...** [S.l.: s.n.], 2014. p.1–6.

COSTA, P.; GUIZZARDI, G.; ALMEIDA, J.; PIRES, L.; SINDEREN, M. van. Situations in Conceptual Modeling of Context. In: ENTERPRISE DISTRIBUTED OBJECT COMPUTING CONFERENCE WORKSHOPS, 2006. EDOCW '06. 10TH IEEE INTERNATIONAL, 2006. **Anais...** [S.l.: s.n.], 2006. p.6–6.

COUTAZ, J.; CROWLEY, J. L.; DOBSON, S.; GARLAN, D. Context is Key. **Commun. ACM**, New York, NY, USA, v.48, n.3, p.49–53, Mar. 2005.

CUPPENS-BOULAHIA, N.; CUPPENS, F.; VERGARA, J. de; VAZQUEZ, E.; GUERRA, J.; DEBAR, H. An ontology-based approach to react to network attacks. In: RISKS AND SECURITY OF INTERNET AND SYSTEMS, 2008. CRISIS '08. THIRD INTERNATIONAL CONFERENCE ON, 2008. **Anais...** [S.l.: s.n.], 2008. p.27–35.

DEY, A. K. Understanding and Using Context. **Personal and Ubiquitous Computing**, [S.l.], v.5, p.4–7, 2001.

DEY, A. K.; ABOWD, G. D. Towards a better understanding of context and context-awareness. In: IN HUC '99: PROCEEDINGS OF THE 1ST INTERNATIONAL SYMPOSIUM ON HANDHELD AND UBIQUITOUS COMPUTING, 1999. **Anais...** Springer-Verlag, 1999. p.304–307.

EKELHART, A.; FENZ, S.; KLEMEN, M.; WEIPPL, E. Security Ontologies: Improving Quantitative Risk Analysis. In: SYSTEM SCIENCES, 2007. HICSS 2007. 40TH ANNUAL HAWAII INTERNATIONAL CONFERENCE ON, 2007. **Anais...** [S.l.: s.n.], 2007. p.156a–156a.

FERRAIOLO, D.; KUHN, R. Role-Based Access Control. In: IN 15TH NIST-NCSC NATIONAL COMPUTER SECURITY CONFERENCE, 1992. **Anais...** [S.l.: s.n.], 1992. p.554–563.

FICCO, M.; ROMANO, L. A Generic Intrusion Detection and Diagnoser System Based on Complex Event Processing. In: DATA COMPRESSION, COMMUNICATIONS AND PROCESSING (CCP), 2011 FIRST INTERNATIONAL CONFERENCE ON, 2011. **Anais...** [S.l.: s.n.], 2011. p.275–284.

GENEIATAKIS, D.; LAMBRINOUDAKIS, C. An ontology description for SIP security flaws. **Computer Communications**, [S.l.], v.30, n.6, p.1367–1374, 2007.

GRUBER, T. R. A translation approach to portable ontology specifications. **Knowledge Acquisition**, London, UK, UK, v.5, n.2, p.199–220, June 1993.

HANSMAN, S.; HUNT, R. A Taxonomy of Network and Computer Attacks. **Comput. Secur.**, Oxford, UK, UK, v.24, n.1, p.31–43, Feb. 2005.

HEERDEN, R. van; LEENEN, L.; IRWIN, B. Automated classification of computer network attacks. In: ADAPTIVE SCIENCE AND TECHNOLOGY (ICAST), 2013 INTERNATIONAL CONFERENCE ON, 2013. **Anais...** [S.l.: s.n.], 2013. p.1–7.

ISOTAMI SEIJI ; BITTENCOURT, I. I. **Dados Aberto Conectados**. first.ed. [S.l.]: Novatec, 2015.

J. PASCOE, N. S. R.; MORSE, D. R. **Human Computer Giraffe Interaction**: HCI in the Field. 182-196p. (GIST Technical Report G98-1).

JENA. **A free and open source Java framework for building Semantic Web and Linked Data applications.** Disponível em: <http://jena.apache.org/>, acesso em dezembro de 2015.

KNAPPMEYER, M.; KIANI, S.; REETZ, E.; BAKER, N.; TONJES, R. Survey of Context Provisioning Middleware. **Communications Surveys Tutorials, IEEE**, [S.l.], v.15, n.3, p.1492–1519, Third 2013.

KOKAR, M. M.; MATHEUS, C. J.; BACLAWSKI, K. Ontology-based Situation Awareness. **Inf. Fusion**, Amsterdam, The Netherlands, The Netherlands, v.10, n.1, p.83–98, Jan. 2009.

KUMAR, J. S.; PATEL, D. R. Article: A Survey on Internet of Things: Security and Privacy Issues. **International Journal of Computer Applications**, [S.l.], v.90, n.11, p.20–26, March 2014. Full text available.

KUMAZAWA, T.; SAITO, O.; KOZAKI, K.; MATSUI, T.; MIZOGUCHI, R. Toward knowledge structuring of sustainability science based on ontology engineering. **Sustainability Science**, [S.l.], v.4, n.1, p.99–116, 2009. An erratum to this article can be found at http://dx.doi.org/10.1007/s11625-009-0076-2.

LASHERAS, J.; VALENCIA-GARCíA, R.; FERNáNDEZ-BREIS, J. T.; TOVAL, A. Modelling Reusable Security Requirements based on an Ontology Framework. **Journal of Research and Practice in Information Technology**, [S.l.], v.41, n.2, p.119 – 133, 2009.

LOPES, J.; SOUZA, R.; GADOTTI, G.; PERNAS, A.; YAMIN, A.; GEYER, C. An Architectural Model for Situation Awareness in Ubiquitous Computing. **Latin America**

Transactions, IEEE (Revista IEEE America Latina), [S.l.], v.12, n.6, p.1113–1119, Sept 2014.

MACHADO, R. S.; ALMEIDA, R. B.; ROSA, D. Y. L. da; LOPES, J. L. B.; PERNAS, A. M.; YAMIN, A. C. EXEHDA-HM: A compositional approach to explore contextual information on hybrid models. **Future Generation Computer Systems**, USA, v.73, p.1 – 12, 2017.

MARTIMIANO, L. A. F.; MOREIRA, E. The Evaluation Process of a Computer Security Incident Ontology. In: THE 2ND WORKSHOP ON ONTOLOGIES AND THEIR APPLICATIONS, RIBEIRÃO PRETO, 2006. **Anais...** [S.l.: s.n.], 2006.

MASSACCI, F.; MYLOPOULOS, J.; PACI, F.; TUN, T.; YU, Y. An Extended Ontology for Security Requirements. In: SALINESI, C.; PASTOR, O. (Ed.). **Advanced Information Systems Engineering Workshops**. [S.l.]: Springer Berlin Heidelberg, 2011. p.622–636. (Lecture Notes in Business Information Processing, v.83).

MONIRUZZAMAN, A. B. M.; HOSSAIN, S. A. NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. **CoRR**, [S.l.], v.abs/1307.0191, 2013.

MOURATIDIS, H.; GIORGINI, P.; MANSON, G. An Ontology for Modelling Security: The Tropos Approach. In: PALADE, V.; HOWLETT, R.; JAIN, L. (Ed.). **Knowledge-Based Intelligent Information and Engineering Systems**. [S.l.]: Springer Berlin Heidelberg, 2003. p.1387–1394. (Lecture Notes in Computer Science, v.2773).

MYLOPOULOS, J.; BORGIDA, A.; JARKE, M.; KOUBARAKIS, M. Telos: Representing Knowledge About Information Systems. **ACM Trans. Inf. Syst.**, New York, NY, USA, v.8, n.4, p.325–362, Oct. 1990.

NOY, N. F.; MCGUINNESS, D. L. **Ontology Development 101**: A Guide to Creating Your First Ontology. [S.l.: s.n.], 2001.

ONWUBIKO, C. Functional requirements of situational awareness in computer network security. In: INTELLIGENCE AND SECURITY INFORMATICS, 2009. ISI '09. IEEE INTERNATIONAL CONFERENCE ON, 2009. **Anais...** [S.l.: s.n.], 2009. p.209–213.

PERERA, C.; ZASLAVSKY, A. B.; CHRISTEN, P.; GEORGAKOPOULOS, D. Context Aware Computing for The Internet of Things: A Survey. **CoRR**, [S.l.], v.abs/1305.0982, 2013.

PERERA, C.; ZASLAVSKY, A. B.; CHRISTEN, P.; GEORGAKOPOULOS, D. Context Aware Computing for The Internet of Things: A Survey. **CoRR**, [S.l.], v.abs/1305.0982, 2013.

PERERA, C.; ZASLAVSKY, A.; CHRISTEN, P.; GEORGAKOPOULOS, D. Context Aware Computing for The Internet of Things: A Survey. **Communications Surveys Tutorials, IEEE**, [S.l.], v.16, n.1, p.414–454, First 2014.

PWC. **Strengthening digital society against cyber shocks, Key findings from The Global State of Information Security Survey 2018**. [S.l.]: PwC International Limited, 2018.

SADALAGE, P.; FOWLER, M. **NoSQL Essencial, Um Guia Conciso para o Mundo Emergente da Persistência Poliglota**. [S.l.]: Novatec, 2013.

SCHMIDT, A.; BEIGL, M.; GELLERSEN, H. w. There is more to Context than Location. **Computers and Graphics**, [S.l.], v.23, p.893–901, 1998.

SOUAG, A.; SALINESI, C.; COMYN-WATTIAU, I. Ontologies for Security Requirements: A Literature Survey and Classification. In: BAJEC, M.; EDER, J. (Ed.). **Advanced Information Systems Engineering Workshops**. [S.l.]: Springer Berlin Heidelberg, 2012. p.61–69. (Lecture Notes in Business Information Processing, v.112).

STRANG, T.; LINNHOFF-POPIEN, C. A Context Modeling Survey. In: IN: WORKSHOP ON ADVANCED CONTEXT MODELLING, REASONING AND MANAGEMENT, UBICOMP 2004 - THE SIXTH INTERNATIONAL CONFERENCE ON UBIQUITOUS COMPUTING, NOTTINGHAM/ENGLAND, 2004. **Anais...** [S.l.: s.n.], 2004.

USCHOLD, M.; GRUNINGER, M. Ontologies: Principles, methods and applications. **KNOWLEDGE ENGINEERING REVIEW**, [S.l.], v.11, p.93–136, 1996.

VOROBIEV, A.; BEKMAMEDOVA, N. An Ontology-Driven Approach Applied to Information Security. **Journal of Research and Practice in Information Technology**, [S.l.], v.42, n.1, p.61–76, 2010.

VOROBIEV, A.; HAN, J. Security Attack Ontology for Web Services. In: SEMANTICS, KNOWLEDGE AND GRID, 2006. SKG '06. SECOND INTERNATIONAL CONFERENCE ON, 2006. **Anais...** [S.l.: s.n.], 2006. p.42–42.

WEISER, M. The Computer for the 21st Century. **Scientific American**, [S.l.], v.265, n.3, p.66–75, January 1991.

YAMIN, A. C. **Arquitetura para um Ambiente de Grade Computacional Direcionada as aplicações Distribuidas, Móveis e Conscientes de Contexto na Computação Pervasiva**. 2004. Tese de Doutorado em Ciência da Computação — Instituto de Informática/UFRGS, Porto Alegre-RS.

YE, J.; STEVENSON, G.; DOBSON, S. A top-level ontology for smart environments. **Pervasive and Mobile Computing**, [S.l.], v.7, n.3, p.359 – 378, 2011. Knowledge-Driven Activity Recognition in Intelligent Environments.